Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Engenharia de Computação e Automação Industrial

CLASSIFICAÇÃO AUTOMÁTICA E ANÁLISE DE DADOS POR REDES NEURAIS AUTO-ORGANIZÁVEIS

JOSÉ ALFREDO FERREIRA COSTA

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação, da Universidade Estadual de Campinas, como parte dos requisitos exigidos para obtenção do título de Doutor em Engenharia Elétrica.

BANCA:
PROF. DR. MÁRCIO LUIZ DE ANDRADE NETTO (ORIENTADOR)
PROF. DR. NELSON DELFINO D'ÁVILA MASCARENHAS
PROF. DR. MAURÍCIO FERNANDES FIGUEIREDO
PROF. DR. FERNANDO ANTÔNIO CAMPOS GOMIDE
PROF. DR. FERNANDO JOSÉ VON ZUBEN
PROF. DR. ROBERTO DE ALENCAR LOTUFO (SUPLENTE)
PROF. DR. RICARDO RIBEIRO GUDWIN (SUPLENTE)

Campinas, SP, dezembro de 1999

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

C823c

Costa, José Alfredo Ferreira
Classificação automática e análise de dados por redes neurais auto-organizáveis / José Alfredo Ferreira Costa.--Campinas, SP: [s.n.], 1999.

Orientador: Márcio Luiz de Andrade Netto.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Sistemas de reconhecimento de padrões. 2. Análise por conglomerados. 3. Redes neurais (Computação). 4. Inteligência artificial. I. Andrade Netto, Márcio Luiz de. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Resumo

Esta tese apresenta extensões ao modelo básico de rede neural auto-organizável, a rede de Kohonen (SOM), viabilizando seu uso como ferramenta de análise de agrupamentos. O SOM define, via treinamento não supervisionado, um mapeamento de um espaço p-dimensional contínuo para um conjunto discreto de vetores referência, ou neurônios, geralmente dispostos na forma de uma matriz. Cada neurônio tem a mesma dimensão do espaço de entrada, p, e o objetivo principal do treinamento é reduzir dimensionalidade ao mesmo tempo em que tenta-se preservar, ao máximo, a topologia do espaço de entrada. O algoritmo SL-SOM (*Self-Labeling SOM*) foi desenvolvido com o objetivo de particionar e rotular automaticamente um SOM treinado, baseando-se no gradiente dos p componentes, cuja informação é apresentada na *U-matrix*. Usa-se algoritmos de processamento de imagem para segmentar a *U-matrix* e o resultado são regiões conectadas de neurônios codificados sob o mesmo rótulo. Tais regiões definem no espaço de atributos geometrias complexas e não paramétricas, possibilitando também a classificação de novas amostras.

A extensão do SL-SOM tem por objetivo descobrir e representar *subclasses*. O TS-SL-SOM (*Tree-Structured Self-Labeling SOM*) gera sub-redes para cada região rotulada de neurônios na forma de uma árvore dinâmica. Não se especifica *a priori* o número de sub-redes para uma dada rede, e os parâmetros de cada sub-rede são funções dos parâmetros da rede 'pai', e do subconjunto de dados que será usado para treiná-la. Sub-redes que não apresentam sub-partições são excluídas, e o conjunto de dados referente àquela sub-rede fica representado apenas pela região rotulada de neurônios na rede 'pai'.

Arranjos de neurônios do SOM de dimensões elevadas não são usados na prática por que o objetivo principal do SOM na atualidade é a visualização dos dados. Com a automação da descoberta de conhecimento e relacionamentos entre dados descritas pelo SL-SOM e TS-SL-SOM, pode-se usar um arranjo dimensão igual ou menor que a dimensão do espaço de entrada, e fazer com que apenas os resultados finais sejam mostrados, na forma de subgrupos de dados, o relacionamento entre os subgrupos, etc. A principal motivação para o uso do SOM p-dimensional é a manutenção da topologia que geralmente é perdida quando diminuímos a dimensionalidade via mapeamento de um espaço p-dimensional para um espaço de menor dimensão. Define-se o *U-array* como uma extensão da *U-matrix* e propõe-se métodos de análise baseados nos métodos de segmentação utilizados em redes de dimensão 1 ou 2.

Comparações de resultados para vários conjuntos de dados são efetuados em relação ao SOM convencional, ou alguns de seus variantes, e por métodos estatísticos e heurísticos para descoberta de agrupamentos, sendo o principal deles, o método de misturas de densidades de probabilidades usando o algoritmo *Expectation Maximization*. As aplicações dos resultados desta tese são inúmeras. Pode-se aplicar técnicas de análise de dados em qualquer área do conhecimento humano que possa coletar informações. Com a disponibilidade crescente de instrumentação eletrônica capacitando aplicações diversas adquirirem dados e armazená-los em computadores, ou mesmo a imensa massa de dados e informações não estruturadas na internet, ferramentas como as descritas nesta tese, com certeza, farão parte de *softwares* em um futuro não distante.

Palavras chaves: *Classificação não-supervisionada; Análise de agrupamentos; Redes neurais auto-organizáveis; Reconhecimento de Padrões; Sistemas inteligentes.*

Abstract

This thesis presents extensions to the most used self-organizing neural network model, the Kohonen network (SOM), enabling its usage as an effective tool for cluster analysis. The SOM network defines, via unsupervised learning, a mapping of a continuos p-dimensional space to a set of model vectors, or neurons, usually arranged as a 2-D array. Each neuron has the same dimension of the input space, p, and the main objective is dimensionality reduction while trying to preserve as much as possible the topology of the input space. The SL-SOM (*Self-Labeling SOM*) algorithm was developed for automatically partitioning and labeling a trained SOM network. It uses information of the p component gradient (distances) which is presented in the U-matrix. By using image processing algorithms, the obtained results are labeled and connected regions of neurons. Each region defines, in the input space, complex and non-parametric geometries which approximately describe the shape of the clusters. Classification of new objects can be performed using the established regions and the nearest neighbor rule.

An extension of the SL-SOM algorithm aims to enhance the clustering process, enabling to discover *sub-clusters*. The TS-SL-SOM (*Tree-Structured Self-Labeling SOM*) algorithm generates a child network for each labeled region of the root network, and so on. The process can be seen as generation of a dynamic tree, where each node is a whole network, and which is data-driven. It is not necessary to specify the number of sub-networks for a given network in a given height of the tree. The parameters of the child network are functions of the parameters of the father network and of the subgroup of data used to train that network. A pruning strategy cuts sub-networks (leave nodes) which do not present further partitions.

High dimension output SOM networks are not frequently used because the main application of SOM is visualization of data in a form of display. With the automation of knowledge discovery and data relations by the SL-SOM and TS-SL-SOM algorithms, we can use output dimensions higher than 2 and analyze only the final results, i.e., number of clusters and their components, relationships between groups, etc. The main advantage of using high dimension output SOMs is that topology preservation is usually lost when mapping a higher input space to a lower output space. The U-array is defined as an extension of the U-matrix and methods are proposed for its segmentation in a similar fashion of those presented in the SL-SOM algorithm.

The thesis also presents results of the methods for synthetic and real data sets, and some comparisons with conventional clustering approaches, such as k-means and mixtures of probability density functions with the *Expectation Maximization* algorithm. Applications of the methods presented in this thesis are numerous. Virtually any area which possess data could be a candidate for using some kind of mapping and thus using any of these methods. With the increasingly availability of masses of data elsewhere, in applications ranging from business to scientific tasks, or even the immense mass of unstructured data available in the internet, and decreasingly cost of memory and computers, tools as the ones presented in this thesis will be important parts of *softwares* in a near future.

Keywords: Unsupervised pattern classification; Cluster analysis; Self-organizing maps; Artificial neural networks; Pattern recognition; Intelligent systems.

Dedicatória

ao meu filho, J. Américo Neto,

e à minha esposa, Bianca.

Agradecimentos

Desejo apresentar minha gratidão e reconhecer o esforço de muitas pessoas que até hoje me ajudaram. Infelizmente o espaço é limitado e esta lista é não – exaustiva.

Inicialmente a *Deus* por me dar saúde e inspiração, e um sentimento de busca por um mundo melhor.

À minha esposa *Bianca* e meu filho *José Américo Neto*, pela paciência, compreensão e amor.

Aos meus familiares, em especial meus pais, *Prof. Dr. José Américo e Gisélia Costa*, e meus irmãos (*Cris*, *Luciana* e *Marcelo*) pelo apoio e incentivo contínuo e incondicional. Também à minha avó, *Zefinha*, e *in memorian* ao meu avô *Dr. Américo Costa*, o meu agradecimento eterno pelas lições valorosas e do sentimento de busca do conhecimento verdadeiro.

Em especial ao amigo e orientador, *Prof. Dr. Márcio L. de Andrade Netto*, pelo incentivo constante e pela amizade solidificada ao longo destes anos.

À CAPES, pelo suporte financeiro na forma de bolsa de estudos.

Ao Departamento de Engenharia de Computação e Automação Industrial (DCA) e a Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, que além de prover um ambiente adequado ao desenvolvimento de pesquisas de nível internacional, também me possibilitou experiências de *Estágios de Capacitação à Docência (PECD)* e de auxiliar didático.

Vários professores e pesquisadores foram muito gentis, muitas vezes tirando dúvidas ou debatendo assuntos de interesse. Exemplos da FEEC/UNICAMP incluem os Professores Doutores *R. Lotufo,* principalmente em assuntos relacionados a morfologia e análise de imagens, *F. von Zuben, F. Gomide* e *R. Gudwin*. Agradeço a vários outros como *N. Mascarenhas e J. Saito* (DC, UFSCar), *R. Moran* (IMECC, UNICAMP), *L. Costa* (IFSC, USP) e *M. Fernandes* (DIn, UEM).

Muitos pesquisadores do exterior também foram gentis, enviando contribuições recentes e mantendo contato pessoal ou através de endereços eletrônicos, entre os quais posso citar: *A. Ultsch* (*Univ. of Marburg*, Alemanha), *T. Kohonen, S. Kaski* e *J. Vesanto* e ao *SOM Toolbox Team* (Helsinki Univ. of Technology, Finlândia), *F. Murtagh* (Univ. of Ulster, UK), *Jim Bezdek* (West Florida Univ.), *Denis Hamad* (Univ. of Lille, França), *C. Lenart* (MIT), *Anil K. Jain* (State Univ. of Michigan), *D. Merkl* (Tech. Univ. of Vienna), *K. Funatsu* (TUT, Japão), *H. Ritter* (Univ. of Bielefeld, Alemanha), entre tantos outros.

Também, à funcionária Carmem (DCA) e aos colegas da pós-graduação em engenharia elétrica da Unicamp, entre os quais *F. Nogueira, Franciraldo Jr.*, *Well*, e *L. Nunes*. A todos estes e tantos outros não incluídos por razões de espaço, muito obrigado pelas amizade, informações, comentários e discussões sobre temas pertinentes do meu trabalho.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Introdução

Vivemos em um mundo complexo, onde estamos sujeitos a uma grande diversidade de estímulos que variam com o tempo. Com certeza seria extremamente difícil representar o conhecimento de todas as coisas de forma diferente, única. Mesmo supondo que todos os estímulos fossem diferentes, parece que a natureza arquitetou um esquema de representações internas destes de forma a agrupá-los em categorias, a qual podemos pensar que é uma generalização de um dado tipo de estímulo, ou um conceito. Categorias de estímulos podem ser formadas percebendo as características essenciais ou comuns aos estímulos. Quando pensa-se em um objeto, por exemplo, um automóvel, imaginam-se características comuns como *'possui quatro rodas'*, *'serve para transporte'*, etc. A essência de uma categoria são os atributos comuns a todos os estímulos que fazem parte dela. O conceito, ou categoria, *'automóvel'* não implica nas particularidades como por exemplo, a cor. Evidentemente as características particulares dos estímulos os diferenciarão dentro da classe, e desta forma, podemos pensar em hierarquias de classes de estímulos e de suas representações internas.

O objetivo básico da formação de categorias de estímulos certamente inclui facilitar a tarefa de identificação (ou associação) de novos estímulos em uma das categorias de estímulos semelhantes, categorias estas formadas em instantes de tempos anteriores. Por outro lado, deve haver um mecanismo que possibilite a incorporação de novos conceitos, o que em neurologia chama-se de *plasticidade neural*. Basicamente, o mecanismo de categorização de estímulos visa ganhar tempo em tomadas de decisões e também na própria formação do pensamento, o qual poderíamos conceber como uma seqüência no tempo de ativações de modelos internos dos estímulos percebidos pelo indivíduo. Assim, a capacidade de formar representações internas eficientes do ambiente e das situações a qual estamos sujeitos é de importância fundamental para a vida e para a adaptação dos seres a um mundo em constantes transformações.

Diariamente uma quantidade imensa de informações são coletadas e armazenadas nos mais diversos arquivos de dados, como por exemplo nos bancos, hospitais, indústrias,

supermercados, laboratórios, etc. Analisar e visualizar grandes volumes de dados na forma de registros, descritos por p atributos, suas inter-relações, similaridades inerentes, etc., torna-se um problema bastante difícil, principalmente pelo fato de que freqüentemente $p >> 3$.

A disponibilidade crescente de dados em sistemas computacionais requer métodos e ferramentas eficientes para sua análise e organização. Classificação automática de dados, ou análise de agrupamentos (*cluster analysis*), são sub-áreas de *análise multivariada*, que por sua vez é uma sub-área da *estatística*, dedicada a análise de problemas onde amostras são descritas por variáveis p-dimensionais. Outra denominação mais geral que tem sido usada é análise exploratória de dados. Na área de *engenharia* tais problemas são referenciados como aprendizado não supervisionado, que é uma sub-área de reconhecimento de padrões. A palavra auto-organização também tem sido utilizada com freqüência na literatura, sobretudo no contexto de redes neurais artificiais, para descrever a busca pela solução de forma não supervisionada.

Como ocorreu em outras áreas do conhecimento humano, o uso de computadores digitais possibilitou grandes avanços nos métodos e nos algoritmos, e poderíamos até dizer que, análise multivariada *prática* só é possível com o auxílio de métodos e recursos computacionais. Recentemente classificação automática e análise de dados têm recebido bastante interesse da comunidade científica, sendo usadas como as ferramentas básicas nas áreas recentes de mineração de dados (*data mining*) e descoberta de conhecimento (*knowledge discovery*). Avanços em instrumentação eletrônica, por outro lado, permitem que dados sejam coletados nos mais variados processos, e transformados em arquivos via interfaces, cujo custo decresce a cada dia. Os campos de aplicação são inúmeros, por exemplo, de processos químicos a exames médicos.

Esta tese discute métodos de classificação automática de dados através de métodos não supervisionados, dando enfoque maior às redes neurais auto-organizáveis. Diferentemente de problemas de reconhecimento de padrões usando treinamento supervisionado, onde um conjunto rotulado de amostras é usado para treinar o classificador, neste trabalho assumimos que a única informação disponível são os dados, i.e., n registros de um banco de dados, descritos por p atributos. O objetivo básico é descobrir a estrutura inerente dos dados, ou o processo de geração destes (no caso estatístico).

Os mapas de atributos auto-organizáveis (ou *self-organizing maps* - SOM), também conhecidos como redes de Kohonen (Kohonen, 1989a), têm sido usados largamente como uma ferramenta de visualização de dados apresentados em dimensões elevadas. O SOM define, via treinamento não supervisionado, um mapeamento de um espaço p-dimensional

contínuo para um conjunto discreto de vetores referência, ou neurônios, geralmente dispostos na forma de uma matriz. Cada neurônio tem a mesma dimensão do espaço de entrada, p, e o objetivo principal do treinamento é reduzir dimensionalidade ao mesmo tempo em que tenta-se preservar, ao máximo, a topologia do espaço de entrada.

O uso do SOM em classificação de dados requer ferramentas adicionais. Em um SOM tradicional, a única informação de saída quando apresentamos um padrão, x, são os índices (i, j) do neurônio vencedor[1], c, e o erro de quantização, que pode ser dado pela distância $d(x, c)$. Geralmente usam-se informações da classe dos padrões mais freqüentes para rotular neurônios em um mapa organizado. Este trabalho busca por soluções automáticas de descoberta de agrupamentos nos dados usando a propriedade de aproximação de densidade de probabilidade do espaço de entrada pelo SOM. O SOM funciona como uma rede elástica ocupando o espaço p-dimensional de forma a representar da melhor maneira, dada uma topologia de vizinhança entre os neurônios, as regiões do espaço com maior densidade de pontos. A visualização das relações entre os neurônios no espaço p-dimensional, em um SOM treinado, é possível através da *U-matrix*, que é uma matriz de distâncias entre os neurônios, com tamanho $(2N\text{-}1) \times (2M\text{-}1)$, onde N e M são as dimensões do SOM. Neurônios vizinhos no mapa e que estejam próximo no espaço p-dimensional terão distâncias pequenas, que corresponderão a 'vales' na *U-matrix*, quando visualizamos a imagem no espaço 3D como um relevo topográfico. As regiões da *U-matrix* com valores elevados correspondem a bordas de regiões de neurônios vizinhos no mapa, e desta forma, a qualidade de sua detecção implica diretamente no resultado dos agrupamentos obtidos. Geralmente a *U-matrix* é uma imagem relativamente complexa, com muitos mínimos locais e outros fatores, sendo sua segmentação um processo não trivial.

A tese apresenta três contribuições principais. A primeira, é o uso de morfologia matemática para segmentar a *U-matrix* de um SOM treinado. O algoritmo *watershed* é aplicado após a mudança da homotopia da *U-matrix* por marcadores que são encontrados após uma análise de estabilidade das regiões conectadas da *U-matrix* para vários níveis de limiarização. Após a segmentação, o número de regiões conectadas reflete o número de agrupamentos presente nos dados e cada região é codificada de forma que todos os seus neurônios possuam o mesmo código. Representar agrupamentos do espaço p-dimensional por vários neurônios implica diretamente na flexibilização da geometria que está sendo descoberta. Apesar das influências de cada neurônio ser isotrópica, a influência global das regiões de neurônios rotuladas no espaço é a integral das influências de seus neurônios, que pode ter uma forma qualquer. Isto permite a descoberta de geometrias variadas de agrupamentos usando o SOM, diferentemente dos métodos estatísticos, que geralmente assumem *clusters* nas formas hiper-esféricas ou hiper-elipsoidais. A este método

denominamos modelo distribuído de protótipos dos agrupamentos. O algoritmo que efetua particionamento e rotulação automática de um mapa do tipo SOM treinado foi denominado SL-SOM (*Self-Labeled SOM*).

A segunda contribuição corresponde a uma extensão do modelo descrito para representar *sub-clusters*. O algoritmo *TS-SL-SOM* (*Tree-structured Self-Labeled SOM*) gera sub-redes para cada região rotulada de neurônios na forma de uma árvore dinâmica. Não se especifica *a priori* o número de sub-redes para uma dado SOM em uma dada posição na árvore, e os parâmetros de cada sub-rede são funções dos parâmetros da rede 'pai', e do subconjunto de dados que será usado para treiná-la. Sub-redes que não apresentam sub-partições são excluídas, e o conjunto de dados referente àquela sub-rede fica representado apenas pela região rotulada de neurônios na rede 'pai'.

A terceira contribuição refere-se à extensão da análise para redes SOM de dimensões elevadas. Define-se o *U-array* como uma extensão da *U-matrix* e propõe-se métodos de análise baseadas nos métodos de segmentação utilizados em redes de dimensão 1 ou 2. SOM de dimensões elevadas não são usados na prática por que o objetivo principal do SOM na atualidade é a visualização dos dados. Como nosso propósito é a automação da descoberta de conhecimento e relacionamentos entre dados, pode-se usar uma rede de dimensão igual ou menor que a dimensão do espaço de entrada, e fazer com que apenas os resultados finais sejam mostrados, na forma de subgrupos de dados, o relacionamento entre os subgrupos, etc. A principal motivação para o uso do SOM p-dimensional é a manutenção da topologia das classes sendo descobertas que é geralmente perdida quando diminuímos a dimensionalidade, i.e., classes vizinhas no espaço p-dimensional podem ficar representadas em um mapa, por exemplo bidimensional, em regiões distantes. Isto ocorre pela 'torção' da rede elástica definida pela vizinhança entre os neurônios. Sendo da mesma dimensão do espaço de entrada, assegura-se a preservação da topologia no mapeamento, e outras informações, tais como as classes que são realmente vizinhas no espaço p-dimensional, podem ser obtidas.

Este trabalho também descreve métodos estatísticos e heurísticos para descoberta de *clusters*, sendo o principal deles, a análise de agrupamentos por misturas de densidades de probabilidades usando o algoritmo *Expectation Maximization*. Os resultados para vários conjuntos de dados são apresentados e discute-se diferenças dos métodos propostos com técnicas estatísticas e com o SOM convencional, e algumas vezes com alguns de seus variantes. As aplicações dos resultados desta tese são inúmeras. Pode-se aplicar técnicas de análise de dados em qualquer área do conhecimento humano que possa coletar informações. Com a disponibilidade crescente de instrumentação eletrônica, capacitando

[1] Assumindo vizinhança bidimensional.

aplicações diversas a adquirirem dados e armazená-los em computadores, ou mesmo com a imensa massa de dados e informações não estruturadas na internet, ferramentas como as descritas nesta tese com certeza farão parte de *softwares* em um futuro não distante.

1.2 Um pouco sobre Auto-Organização

Devido a esta tese descrever e usar modelos de redes neurais artificiais denominados auto-organizáveis, esta seção aborda brevemente tópicos relacionados ao assunto, situando o leitor no que a palavra auto-organização está associada a estes modelos.

Processos auto-organizados têm sido estudados nas mais diversas áreas (Debrun et al., 1996). Geralmente o conceito de auto-organização (AO) está associado ao aumento da organização (ou ordem) de um sistema sem que o princípio organizador seja um agente externo ao sistema ou um elemento privilegiado dentro dele (Pessoa, 1996). Porém, a caracterização do que seja um processo auto-organizado é muito complexa e ainda é uma questão aberta. Algumas das características básicas dos processos auto-organizados são:

1. Processos auto-organizados são processos coletivos, onde unidades que fazem parte deste coletivo competem, com chances de sucesso semelhantes, por recursos limitados. Chances semelhantes implicam na inexistência de hierarquias ou de elementos privilegiados.

2. O processo é parcialmente autônomo em relação às suas condições iniciais e se desenvolve através de um trabalho de si sobre si.

3. A forma final não é resultante passiva do processo, tem uma identidade, porém não é determinística.

4. Flutuações nos estados dos elementos podem gerar uma força assimétrica que leve os elementos a um estado de maior organização.

Questões não respondidas incluem a própria existência da vida. Como definir a palavra *organização* implicará em possíveis fatores para podermos quantificar o quanto houve de mudança nos estados dos elementos de um sistema em um dado período de tempo. Como o universo está em constante aumento de desordem (o aumento da entropia é crescente), nenhum sistema passivo poderia se auto-organizar (aumentar sua neguentropia). Alguns autores consideram que um sistema se auto-organizou, em um dado grau, se ele aumentou

sua complexidade. Porém, esta complexidade foi devida a diminuição da desordem interna ou transferida pelo mundo externo?

Alguns autores, por exemplo Debrun (1996), diferenciam AO em primária e secundária. A primeira sendo mais forte, com mais independência das condições iniciais do que a segunda, que seria uma variação no nível de complexidade do sistema. AO primária pode ser vista como um processo que gera uma forma que não parte de uma forma inicial, enquanto que AO secundária seria uma adaptação da forma existente, uma acomodação interna ou devida a um estímulo externo.

Como classificar processos em AO primária ou AO secundária? Como quantificar aumento de organização em um sistema? Questões como estas são bastante difíceis e não restringem-se à filosofia. Vários autores, incluindo os da área de cibernética, analisaram o problema de forma sistêmica, inclusive gerando índices para que pudessem comparar diferentes estados de organização. Algumas destas medidas foram descritas em (Pessoa, 1996). Uma das dificuldades é que problemas de diferentes áreas geram noções de organização diferentes e por conseqüência critérios quantitativos e qualitativos diferentes.

A existência da AO primária também tem sido questionada: o que seria um estado do sistema totalmente desorganizado e como um sistema em tais condições teria iniciativa, autonomia e energia para atingir um estado de maior organização? Como podemos pensar em sistemas como conjuntos de sub-sistemas ou partes interconectadas com o objetivo de realizar alguma função ou objetivo, e que estes sub-sistemas foram anteriormente organizados por outros sub-sistemas, e assim por diante, o único organizador primário de todas as coisas seria Deus. Este é um dos pilares da prova lógica da existência de Deus construída há mais de um milênio por São Tomás de Aquino: tudo que existe teve que ser organizado e isto implica em um organizador. Mas o organizador também teve que ser organizado, e assim por diante, e nos leva ao organizador original, o qual haveria de existir desde a eternidade.

Filosoficamente, o conceito de um sistema auto-organizável é tão importante pelo fato de que podemos pensar que toda a vida existente pode ter surgido a partir de um processo auto-organizável, no qual diferentes elementos químicos se juntaram de forma mais ou menos probabilística e que gradualmente se organizaram em padrões vivos e reprodutíveis. A existência de flutuações (ruído) entre os elementos modificam o sistema assimetricamente e o conduz a um outro estado de organização. Caso haja influências externas atuando nesta força assimétrica interna, o sistema pode estar em um processo de adaptação ao ambiente. Caso contrário, podemos pensar como uma acomodação dos elementos, e esta assimetria é considerada uma das bases da auto-organização.

A área de auto-organização em sistemas dissipativos tem sido estudada há vários anos (Nicolis e Prigogine, 1989). O princípio básico é que apenas quando um sistema não linear é colocado longe de seu estado de equilíbrio que ele começa a apresentar o comportamento de auto-organização. Nesta linha de raciocínio, os requerimentos para que AO ocorra são que o sistema esteja aberto a seu ambiente e que haja uma troca contínua de energia e entropia entre eles (Jantsch, 1980). O sistema desordenado está longe do equilíbrio. Por exemplo, um sistema termodinâmico que contém um grande número de elementos interagentes, e sujeitos a restrições não lineares, e onde há presença de flutuações (ruído). Apesar de Prigogine conceber uma transferência de neguentropia do ambiente para o sistema, a mudança do sistema não é induzida apenas por um parâmetro externo, mas é a dinâmica interna do sistema, i.e., a presença de forças tanto atrativas quanto repulsivas, que podem levar a um estado mais ordenado.

Antes disso, Ashby (1962) descreveu o aumento de organização como condicionalidade. Sistemas isolados não poderiam se auto-organizar, e a organização do sistema seria uma propriedade subjetiva, i.e., depende de como o sujeito olha o sistema. Ashby também considera a relação complexidade e adaptabilidade: sistemas isolados suficientemente grandes desenvolvem espontaneamente sub-sistemas bem adaptados que preservam suas propriedades diante de flutuações do ambiente. Para sistemas determinísticos, Ashby argumenta que, em princípio, a maioria dos sistemas dinâmicos vão para estados de equilíbrio. Porém a maioria dos estados dos sistemas são instáveis. Assim, indo de um estado qualquer para um estado de equilíbrio, o sistema está efetuando um processo seletivo, indo de uma classe de estados, onde há grande número destes, para uma classe com menor número de estados.

O conceito de auto-organização em redes neurais artificiais está bastante ligado ao conceito de aprendizado não supervisionado na área de reconhecimento de padrões. Em contraste com os sistemas supervisionados, que são treinados a partir de conjuntos rotulados de padrões, onde o rótulo explicita a categoria do padrão que está sendo inserido no sistema, a única informação utilizada nos sistemas não supervisionados, ou auto-organizáveis, são os próprios padrões. Relações entre padrões devem ser determinadas automaticamente. Por outro lado, devemos diferenciar os objetivos nos dois casos: no treinamento supervisionado, o objetivo é maximizar a generalização do sistema na fase de execução em relação a novos padrões não vistos na fase de aprendizagem. Isto implica em capacidade do sistema ter absorvido corretamente, na fase de treinamento, um comportamento coerente com o conjunto de padrões de treinamento, e que este conjunto seja de fato representativo da população. Por outro lado, no caso de treinamento não supervisionado, o objetivo é buscar a estrutura desconhecida do conjunto de padrões (i.e., das informações). Em vez de

ajustar os parâmetros com o objetivo de minimizar um erro computado a partir de informações *a priori*, relações implícitas de semelhanças e dessemelhanças devem ser obtidas a partir de competição entre as unidades (neurônios).

Motivações biológicas para processos auto-organizados nos sistemas neurais são diversas. Inclui, por exemplo, o fato de que seria impossível armazenar no DNA a informação de todas as conexões a serem feitas entre todos os neurônios do cérebro. Existem cerca de 100 bilhões de neurônios no cérebro humano, cada um conectado em média a outros 10^4 neurônios, resultando em cerca de 10^{15} conexões sinápticas. Apesar de que grande parte da '*programação*' neural do cérebro seja transmitida geneticamente, principalmente concernente às macro-estruturas, existe um processo intenso de criações e eliminações de conexões, principalmente na fase inicial da vida, na qual há formação e especialização de vários mapas sensoriais. A figura 1.1 apresenta três estágios de desenvolvimento da arborização dendrítica no córtex visual humano. Da esquerda para a direita temos a ilustração de um corte perpendicular do córtex de um recém-nascido, com três meses e finalmente dois anos, respectivamente, onde os símbolos I, II, ..., representam as áreas do córtex visual *V* I, *V* II, e assim por diante. Muitos dos modelos internos a respeito do mundo externo assemelham-se a mapas topográficos ordenados, onde estímulos semelhantes ativam centros de ativação próximos, o que ocorre também para classes de estímulos semelhantes. Os mapas são estabelecidos na fase inicial da vida através de mecanismos internos. Porém, a presença de estímulos é de crucial importância. Experiências com animais comprovaram que a ausência de estímulos, por exemplo visuais, nas primeiras semanas pós-parto, comprometem definitivamente a funcionalidade do sistema visual, apesar de tanto a sensitividade da retina quanto dos nervos ópticos que transmitem as percepções ao córtex estarem perfeitas. Mesmo que o animal seja exposto futuramente a grande diversidade de estímulos ópticos a conectividade entre os neurônios do córtex visual ainda permanecerão bastante deficientes. Mais detalhes sobre o funcionamento e modelos neuro-fisiológicos do cérebro podem ser vistos em Zeki (1993), especialmente no que diz respeito aos mapas de visão cromática.

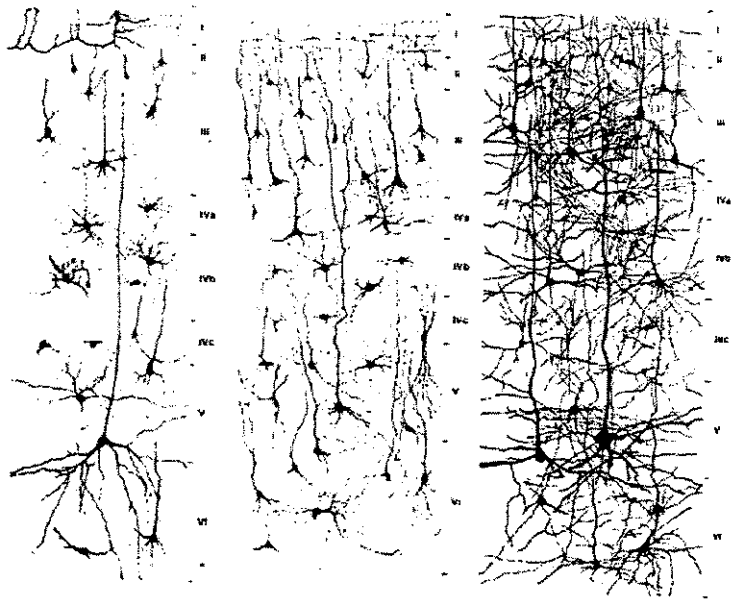


Figura 1.1: Estágios do desenvolvimento da árvore dendrítica no córtex visual humano. Da esquerda para a direita: recém-nascido, com três meses, e com dois anos de idade. Adaptado de Bullock et al. (1977)

No caso particular do treinamento do SOM via auto-organização, temos que o sistema é aberto e sujeito a flutuações externas, que a estrutura da rede é tal que forças atrativas / repulsivas estão presentes quando pesos sinápticos dos neurônios são adaptados de acordo com os padrões apresentados na camada de entrada. O estado organizado resulta das assimetrias nestas forças causadas pela função da vizinhança entre os elementos. A auto-organização está ligada à adaptação do sistema aos estímulos apresentados. A auto-organização estará presente nesta adaptabilidade do sistema em 'distribuir' seus elementos de modo a preservar relações topológicas do espaço de entrada. Esta distribuição também leva em conta a densidade de probabilidades deste espaço, fazendo com que regiões com maior concentração de padrões sejam melhor representadas. Porém, vários parâmetros devem ser corretamente estabelecidos para que obtenhamos as características desejadas, ou que o algoritmo convirja para uma solução adequada. Neste processo auto-organizado de quantização do espaço p-dimensional por elementos que mantêm relações de vizinhança previamente estabelecidas, podemos chegar a soluções totalmente adversas do ideal, que foram denominados de meta-estados (Erwin *et al.*, 1992a,b). Apesar do algoritmo do SOM ser relativamente simples, suas propriedades matemáticas, principalmente no que se refere ao processo de auto-organização, por exemplo para mapas bidimensionais, ainda não foram formalmente demonstradas.

1.3 Organização do trabalho

Esta tese é apresentada em oito capítulos, onde busca-se inicialmente apresentar o problema de classificação automática de padrões e motivações para o uso de técnicas de redes neurais para resolvê-lo. As contribuições originais concentram-se, principalmente, nos capítulos 5-7.

O capítulo 2 apresenta uma breve introdução aos métodos de classificação automática e análise de agrupamentos. Seções incluem métodos hierárquicos e particionais, métodos baseados em lógica nebulosa, misturas de funções densidades de probabilidades com estimação de parâmetros pelo algoritmo *expectation-maximization*, além da idéia que busca-se flexibilizar a geometria dos agrupamentos encontrados, que é o modelo distribuído dos protótipos, onde há possibilidade de uso de vários protótipos representando a estrutura de um agrupamento.

O capítulo 3 descreve o modelo de rede neural auto-organizável que é a base para várias análises nesta tese, o mapa auto-organizável de Kohonen (SOM). São descritos os algoritmos convencional e em lote, este último bastante usado neste trabalho. A superfície de influência dos neurônios é apresentada e mostra-se que concentrando atenção nos neurônios que estão agrupando mais padrões pode-se chegar a conclusões como o número de agrupamentos presente nos dados.

No capítulo 4, alguns modelos de redes neurais competitivas e alguns modelos derivados do SOM são abordados. Muitas das motivações relacionadas a modelos variantes do SOM incluem a flexibilização de escolhas de parâmetros, como o número de neurônios que devem ser inseridos no início do treinamento, no algoritmo convencional. As vantagens e limitações de alguns modelos importantes na literatura recente são descritas, e algumas destas deficiências são motivações para o desenvolvimento dos algoritmos apresentados nos capítulos 5 a 7.

O capítulo 5 apresenta o método da *U-matrix* para visualização da estrutura do SOM. O Método de segmentação de imagens *Watershed*, baseado em morfologia matemática, é descrito e aplicado à *U-matrix*. O algoritmo SL-SOM gera conjuntos de neurônios rotulados que representam os modelos distribuídos dos agrupamentos. Vários exemplos ilustram a eficácia do algoritmo proposto, inclusive usando-se classes não linearmente separáveis[2].

[2] Todas as implementações nesta tese foram feitas usando o Matlab 5.0 ou linguagem C. O algoritmo *watershed* usado foi obtido do *MMorph Toolbox*.

A extensão do método abordado no capítulo 5 é apresentado no capítulo 6, onde uma estrutura hierárquica é apresentada com o objetivo de encontrar, caso existam, subgrupos dentro dos agrupamentos descobertos pelo SL-SOM, a qual denominamos de TS-SL-SOM. Cada grupo de neurônios sob o mesmo rótulo gera um novo mapa e este processo pode ser visto como uma especialização ou focalização da atenção nas partições obtidas nos mapas em níveis superiores da árvore. Os mapas filhos são treinados apenas com o subconjunto de dados relacionado com as regiões rotuladas e relacionadas a eles no mapa *pai*. A hierarquia de sub-redes é definida dinamicamente, de acordo com as informações obtidas de grupos nas redes dos níveis anteriores. Exemplos são descritos e comparações com outros modelos são analisados.

O capítulo 7 apresenta a extensão da análise de agrupamentos em um SOM bidimensional para uma rede SOM com dimensão arbitrária. Apresenta-se a extensão da *U-matrix* como o *U-array*. Propõem-se métodos de análise automática do *U-array* com o objetivo da manutenção da topologia dos dados e das classes, que geralmente é perdida quando mapeamos um espaço de dimensão mais elevada em um espaço de saída de menor dimensão. Resultados são apresentados e comentados.

Finalmente, o capítulo 8 apresenta conclusões e futuras linhas a serem investigadas. Em geral, os métodos de classificação automática e análise de agrupamentos requerem a escolha *a priori* de vários parâmetros que em geral possuem grande influência no resultado final. Vantagens do uso do modelo de protótipos distribuídos e sua implementação via redes SOM são comentados, assim como suas limitações.

1.4 Sumário

Este capítulo teve o objetivo de introduzir o leitor no contexto dos temas abordados na tese. A organização do trabalho foi apresentada, e uma breve discussão sobre auto-organização foi introduzida.

O Matlab® é propriedade da MathWorks, Inc. Endereço eletrônico: http://www.mathworks.com.
O *MMorph Toolbox* é propriedade da SDC. Endereço eletrônico: http://www.sdc.com.

Capítulo 2

Métodos de Classificação Automática de Dados

O objetivo deste capítulo é descrever brevemente alguns métodos de classificação automática de dados, seja por abordagens de aprendizado não supervisionado, modelos de misturas de densidades ou por métodos heurísticos de agrupamentos. Discute-se a importância das várias escolhas no processo de tratamento e representação dos dados, como a importância dos atributos, o critério de similaridade, etc. Apresenta-se a idéia de representação distribuída de protótipos para agrupamentos, o que permite flexibilidade na geometria dos classes a serem descobertas.

2.1 Introdução

Diversas denominações foram atribuídas ao que chamamos *classificação automática* (CA)[1] de dados. Estatísticos geralmente usam o termo *análise de agrupamentos* (do inglês *cluster analysis*). Outras denominações incluem taxonomia numérica, Q-análise e tipologia. Em engenharia, a classificação automática de dados está ligada ao ramo de reconhecimento de padrões que usam métodos não supervisionados para estimar parâmetros de modelos de sistemas que objetivam, inicialmente, descobrir a estrutura de um determinado conjunto de dados não rotulados, $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, onde cada objeto (ou amostra) $\mathbf{x}_i$, $i = 1, \ldots, n$, é descrito por p variáveis (atributos ou características), veja a tabela 2.1. Técnicas de CA podem ser aplicadas virtualmente à qualquer área do conhecimento humano, tais como medicina, psicologia, arqueologia, inteligência artificial, sociologia, biologia, etc. (Everitt, 1993) com o objetivo de prover uma descrição ou síntese dos dados.

Devido à crescente disponibilidade de grandes massas de dados armazenados em computadores, a necessidade de métodos que possam analisá-los de forma automática, ou não supervisionada, torna-se cada vez maior. Aplicações podem ter diferentes objetivos, como por exemplo, a determinação de objetos que sejam semelhantes ou o enfoque em uma determinada classe de objetos. Pode-se fazer uma síntese do banco de dados observando os

[1] Termo mais apropriado, na visão do autor, baseado no termo em francês *"Classification Automatique"*, ver por exemplo (Jambu, 1978; Jambu e Lebeaux, 1983), e no termo em alemão *"Automatische Klassification"* (Bock, 1974).

objetos representantes de cada subgrupo, que vai confirmar, ou não, hipóteses a respeito da massa de dados em questão. Pode-se também formular hipóteses sobre a estrutura dos dados e determinar esquemas de classificação.

Perguntas freqüentes incluem: (1) Existem subgrupos menos heterogêneos nos dados?; (2) Quantos subgrupos de fato existem (se é que existem)?; (3) Que objetos fazem parte de cada subgrupo?; (4) Uma vez encontrado um modelo para a estrutura dos dados poderíamos gerar regras de decisão que possibilitassem a classificação de novas amostras?; Como poderíamos tratar tipos de variáveis diferentes, ex. binárias e contínuas, em cálculos de índices de similaridade?

TABELA 2.1: MATRIZ DE DADOS

	Variáveis			
Objetos	$x_{\cdot 1}$	$x_{\cdot 2}$	...	$x_{\cdot p}$
1	x_{11}	x_{12}	...	x_{1p}
2	x_{21}	x_{22}	...	x_{2p}
:	:	:	...	:
:	:	:	...	:
n	x_{n1}	x_{n2}	...	x_{np}

Estas e tantas outras perguntas, aparentemente simples, tornam-se incrivelmente complexas à medida que os valores de n e p aumentam. Geralmente iremos supor que, a menos que se explicite o contrário, $\boldsymbol{X} \subset \Re^p$, e $C^i \cap C^j = \varnothing$, ou seja, cada objeto do nosso conjunto de dados pode ser visto como um ponto no espaço p-dimensional dos números reais, e que a interseção das partições a serem encontradas pelos algoritmos é vazia[2]. Os agrupamentos, C^i, $i = 1, \ldots, K$, onde K é o número de agrupamentos ou partições, são subconjuntos de $\boldsymbol{X}$ e cada $C^i \neq \varnothing$, i.e., qualquer agrupamento possuirá no mínimo 1 elemento. Uma última característica é que a união dos vários agrupamentos é o próprio conjunto de dados, $\bigcup_{k=1}^{K} C^k = \boldsymbol{X}$. Estas características diferenciam uma coleção arbitrária de pontos (um grupo) de um agrupamento, no qual é um conjunto de pontos ou objetos que também devem possuir uma forte relação de similaridade.

[2] Uma extensão natural do trabalho será o relaxamento desta segunda condição, permitindo a ocorrência de partições nebulosas (Kandel, 1982, 1986), (Pedrycz, 1990), (Bezdek e Pal, 1992).

As variáveis $x_{.j}$ na matriz de dados podem ser quantitativas (discretas ou contínuas) ou qualitativas (ordinais ou nominais). Em muitas situações, podemos ter uma matriz de dados composta de uma mistura de diferentes tipos de variáveis. A dificuldade em encontrar boas soluções na prática pode ser agravada quando temos uma matriz de dados apresentando falta de informações (*missing data*), ou valores discrepantes (*outliers*) ou ainda sobreposição entre os subgrupos (*cluster overlap*).

Um dos problemas encontrados no estudo de CA é a falta de definições formais dos termos envolvidos. A regra básica poderia ser descrita como *'agrupe dados em subgrupos de forma que os elementos constituintes de um subgrupo sejam mais similares entre si do que qualquer outro elemento alocado para outro subgrupo'*. O primeiro problema seria então a escolha do critério de similaridade, que irá depender do problema em questão. Termos como similaridade, agrupamentos naturais, etc., possibilitam muitas interpretações.

A idéia de agrupamento discutida nesta tese baseia-se na definição apresentada em Everitt (1993), onde considera-se que os objetos são pontos no espaço p-dimensional e cada variável representa um eixo neste espaço. Um agrupamento é uma região deste espaço, contínua, que contenha uma densidade de pontos relativamente elevada, separada de outras regiões densas por regiões com baixa densidade de pontos. Cormack (1971) levanta ainda a hipótese de que agrupamentos deveriam possuir coesão interna e isolação externa. Mesmo com tais considerações, há de se comentar que não existe uma única definição para agrupamentos devido às muitas características particulares de cada ramo do conhecimento onde podemos aplicar tais técnicas. É comum o fato do pesquisador conhecer o processo que originou os dados, o que lhe permite escolher um método que adeqüe, de forma satisfatória, sua estrutura geométrica à estrutura esperada dos dados. O problema é que, caso a escolha tenha sido feita de forma inadequada, ainda assim o método irá gerar resultados, mesmo impondo uma nova estrutura aos dados, em vez de recuperar a original.

A escolha, muitas vezes *a priori*, do número ideal de subgrupos, K, é uma das mais importantes. Supondo ter escolhido o valor correto para K, haveria formas de escolher uma partição ótima dos dados, C^k, $k = 1, 2, \ldots K$?

A busca exaustiva pela partição ótima é proibitiva para valores relativamente pequenos de n. Existem aproximadamente $k^n/k!$ possíveis formas de particionar n objetos em k subgrupos. O número exato é dado por análise combinatória: o número de Stirling do segundo tipo (Anderberg, 1973):

$$\varsigma_n^{(k)} = \frac{1}{k!}\sum_{i=0}^{i=k}(-1)^{k-i}\binom{k}{i}i^n . \qquad (2.1)$$

Para n e k iguais a *25* e *5*, respectivamente, $\varsigma_{25}^{(5)} \approx 2.44\times10^{15}$. Quando o número de subgrupos é desconhecido o número de possibilidades é ainda maior, pois é uma soma de números de Stirling. Para n igual a *25*,

$$\sum_{j=1}^{j=25}\varsigma_{25}^{(j)} > 4\times10^{18}$$

que é um número excessivamente grande para um número pequeno de objetos. Na prática, podemos ter valores de n múltiplos de milhões, por exemplo, em um banco de dados de uma instituição financeira. Mesmo dispondo do mais sofisticado sistema computacional disponível, seria impossível, na atualidade, encontrar a solução ótima por meio de um método de busca exaustiva. Desta forma, deve-se encontrar meios eficientes de busca por partições, mesmo que não ótimas, porém que satisfaçam os requisitos da análise.

As figuras 2.1 e 2.2 ilustram o problema de classificação automática de dados. Supondo que cada objeto apresentado na figura 1 representa um registro de um banco de dados, que podem estar bastante misturados, o objetivo básico seria agrupar objetos semelhantes em posições próximas, assim como as classes. Supõe-se já dispor dos p valores para cada um dos n objetos, os quais devem ser escolhidos de forma criteriosa. A escolha de atributos é uma decisão extremamente importante nos processos de reconhecimento de padrões, pois todos os passos seguintes usam esta informação, que representa uma grande compressão de dados em relação ao objeto original. Idealmente escolheríamos atributos que representassem bem as características das classes as quais estamos interessados, que tivessem pouca variância dentro dos objetos da classe, e grandes diferenças entre classes distintas. A escolha é em geral dependente do problema, e mesmo tendo dedicado um bom tempo à escolha de bons atributos, não estamos imunes a outros vários problemas como sobreposição de classes e erros devido a problemas de medição e instrumentação, por exemplo, erros sistemáticos (relacionados à calibração de instrumentos), ou outros erros devido a interferências nas medidas, problemas de leitura ou armazenamento dos dados, etc. Cada objeto representado na figura 2.1 poderia ser visto como, por exemplo, um registro de um exame médico, ou o resultado de uma análise química, ou informações de um sistema de telemetria.

Idealmente, o resultado do agrupamento seria semelhante ao apresentado na figura 2.2, onde objetos similares estão representadas em posições próximas. Na prática, dificilmente

conseguiríamos uma separação ótima, devido, por exemplo, a alguns dos problemas citados no parágrafo anterior. Porém, em geral estaremos satisfeitos em encontrar partições no conjunto de dados X que nos permitam compreender a estrutura inerente aos dados.

Um outro aspecto que seria interessante preservar nos sistemas de classificação automática seria manter classes semelhantes próximas. Esta informação pode ser útil em análises posteriores, em problemas de mineração de dados e descoberta de informações. A figura 2.3 ilustra uma das possíveis partições do espaço de atributos para o problema apresentado, na hipótese de que tal espaço tivesse dimensão 2. As classes 1 e 2, que têm um grau de semelhança maior, do que por exemplo as classes 1 e 5, aparecem próximas na representação final. Idealmente, classes mais distintas deveriam ser representadas em porções mais distantes na partição do espaço de atributos. Porém, isto nem sempre ocorre, na prática, devido à própria escolha dos atributos, que pode ter sido feita de forma não ótima (por exemplo, as classes 4 e 5, que estão vizinhas na figura 2.3), e também pela perda da informação topológica, que ocorre geralmente quando mapeamos um espaço de dimensão maior em um espaço de dimensão menor. A preservação da topologia será a motivação básica para o uso de mapas com dimensão no espaço de saída maior que 2 (capítulo 7).

Deve-se diferenciar *classificação automática* da classificação de padrões obtida por um processo de treinamento supervisionado. Nesta última, o classificador é visto como um discriminador, ou máquina de decisão, a qual foi treinada com dados rotulados, i.e., com origem de classe conhecida, com o objetivo de maximizar a generalização para futuras amostras de dados não usadas durante o treinamento. Por outro lado, classificação automática objetiva descobrir as relações entre os dados, a estrutura destes, e caso seja possível, um esquema de análise de futuras amostras.

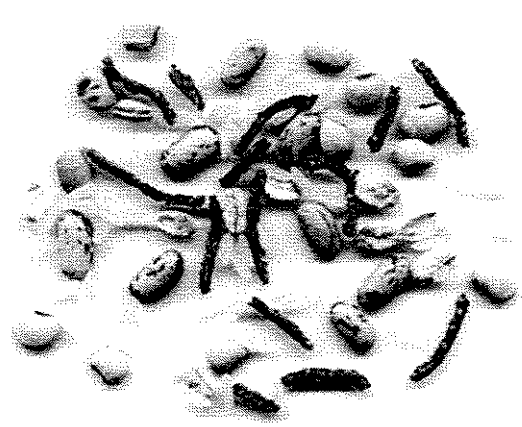


Figura 2.1 - Ilustração representando objetos de vários tipos misturados.

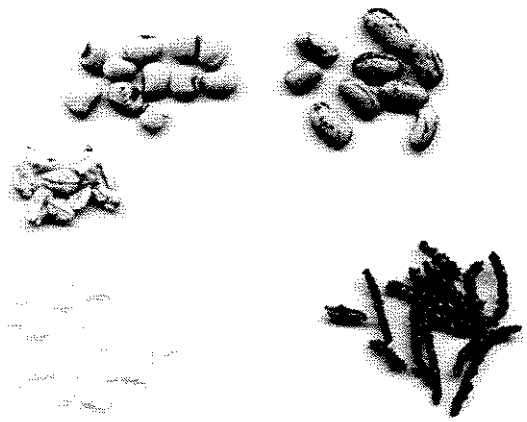


Figura 2.2 - Ilustração representando objetos agrupados: maior homogeneidade dentro dos subgrupos.

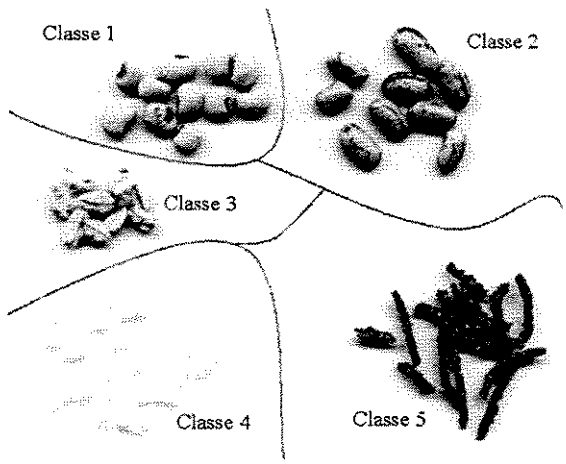


Figura 2.3 - Detecção de classes após o agrupamento efetuado na figura 2.2.

2.2 A literatura em classificação automática

Métodos de classificação são importantes em qualquer área do conhecimento humano e é um processo básico em qualquer ciência. A disponibilidade de recursos computacionais, como *hardware* com baixos custos, e *softwares*, muitos dos quais pacotes de métodos de análise multivariada, incluindo várias rotinas de análise de dados e classificação, têm possibilitado o aumento explosivo de aplicações nas mais variadas áreas. Seria praticamente impossível, atualmente, gerar um artigo representativo da área sem que este contivesse algumas centenas de referências. Artigos gerais descrevendo a área como

Blashfield e Aldenderfer (1978) ou Scoltock (1982) há muito não são escritos, pela grande variabilidade dos métodos e aplicações[3]. Desta forma, não temos a preocupação de sermos exaustivos nesta seção, sendo apenas um breve panorama da literatura da área.

A maioria das publicações estão espalhadas em uma enorme quantidade de jornais de áreas muito diferentes, havendo, no momento, apenas uma publicação dedicada exclusivamente à área, o *Journal of Classification*. Diferentes nomenclaturas e redundância em métodos desenvolvidos aparentemente de forma independente são comuns em CA. Além disto, grande parte dos métodos carecem de formalidades matemáticas. A maioria dos trabalhos publicados na área de CA poderiam ser categorizados em duas grandes classes de métodos, os algoritmos hierárquicos e os não hierárquicos (ou particionais). A figura 2.4 ilustra, de forma bastante simplificada, uma taxonomia dos métodos de classificação automática.

Ambas as classes de métodos possuem domínios apropriados de aplicações. Técnicas hierárquicas são bastante populares nas áreas biológicas, sociais e psicologia, devido à necessidade de se construir taxonomias (ver seção 2.4). Técnicas particionais (seções 2.5-2.7) são usadas com freqüência maior em engenharia onde soluções com uma única partição são importantes (Jain & Dubes, 1988), principalmente quando busca-se representações eficientes e compressão de grandes bases de dados.

Os métodos particionais poderiam ser subdivididos ainda em exclusivos e não exclusivos. Nos métodos exclusivos cada objeto ou item é atribuído a apenas um agrupamento, enquanto que nos métodos não exclusivos, há possibilidade de pertinência a mais de um agrupamento, como por exemplo o *fuzzy k-means* (ver seção 2.6). Exemplos dos métodos particionais exclusivos são o *k-means* (seção 2.5) e os métodos particionais que baseados em misturas de densidades de probabilidades (ver seção 2.7), os quais serão utilizados para testes nesta tese.

O termo *cluster analysis* foi usado pela primeira vez por R. Tryon em 1939, um livro posteriormente reeditado (Tryon & Bailey, 1970), porém apenas no final dos anos 50 os métodos passaram a ser mais pesquisados, inicialmente nas áreas de biologia (Sokal & Sneath, 1973; Cole, 1969), psicologia (Lorr, 1983; Tryon e Bailey (1970)). Vários livros foram publicados nos anos 70, dos quais destacam-se Anderberg (1973), ainda bastante citado atualmente, principalmente pelos ótimos capítulos sobre similaridades e dissimilaridades, Duran e Odell (1974), que deram um enfoque matemático a CA, Everitt (1974) cuja terceira edição Everitt (1993) representa uma das melhores introduções à área,

[3] Em recente correspondência ao autor, Prof. Mark Aldenderfer, da Universidade da Califórnia em Santa Barbara, disse desconhecer qualquer artigo recente sobre a literatura de CA.

e Hartigan (1975), que da mesma forma como Anderberg (1973) ainda é bastante citado e igualmente possui seções com códigos em Fortran dos algoritmos abordados.

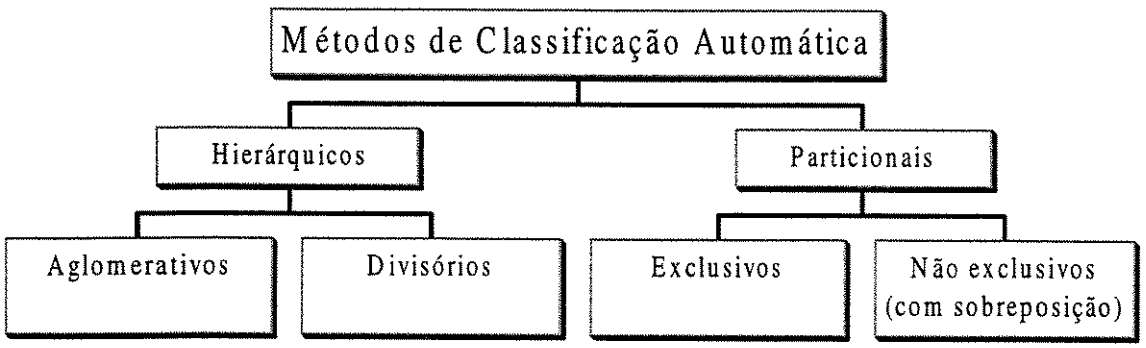


Figura 2.4 - Taxonomia (simplificada) dos métodos de classificação automática de dados

Duda e Hart (1973), recentemente atualizado em Duda *et al.* (1998) mesmo não sendo específicos em CA, tiveram grande repercussão devido a abrangência matemática e explanação de idéias e métodos, possuindo um excelente capítulo dedicado a aprendizado não supervisionado. Livros importantes na década de '80 incluem Spath (1980a), Jain e Dubes (1988), e Massart e Kaufman (1983), este último uma das melhores introduções à área. Livros dedicados a métodos de misturas de densidades de probabilidades foram editados naquela década, como o Titterington *et al.* (1985) e o McLachlan e Basford (1988), os quais usam o método *expectation-maximization* (EM) para estimar os parâmetros das densidades componentes via maximização da função verossimilhança. Os livros recentes mais importantes incluem Kaufmann e Rousseeuw (1990), Everitt (1993) e Arabie *et al.* (1996), este último uma excelente coletânea de *surveys* das várias áreas de CA.

Artigos de revisão importantes incluem Cormack (1971), que analisou de forma bastante crítica alguns métodos de CA, Dubes & Jain (1980) cujo conteúdo está inserido em Jain & Dubes (1988), Diday e Simon (1978) e Hansen e Jaumard (1997). Surveys dedicadas a técnicas hierárquicas incluem Day e Edelsbrunner (1984), Murtagh (1983) e Gordon (1987), este último uma das melhores referências em métodos hierárquicos, o qual foi recentemente adaptado (Gordon, 1996). Milligan (1980) e Milligan and Cooper (1985) apresentaram estudos e análises de métodos hierárquicos fazendo simulações de Monte Carlo. Seus resultados implicam em uma vasta gama de heurísticas para determinação do número adequado de agrupamentos em métodos hierárquicos. Artigos recentes dedicados a métodos particionais, focalizando inclusive aplicações de mineração de dados, incluem Zaït e Messatfa (1997) e Michaud (1997). Artigos com enfoque mais estatístico incluem

MacQueen (1967), Wolfe (1970), Scott and Symons (1971), Hartigan (1977; 1978; 1981; 1985), Symons (1981), Everitt (1981), Bock (1985), e Thode et al. (1988). Métodos Bayesianos também foram abordados, por exemplo em Binder (1978; 1981), Banfield e Raftery (1993), e Bensmail et al. (1997). Técnicas de classificação nebulosas (fuzzy clustering) foram descritas em Bezdek (1981), Kandel (1982) e em Bezdek e Pal (1992a,b), este último uma coletânea de artigos representativos da área.

Técnicas de CA foram aplicadas em uma grande variedade de problemas. Hartigan (1975) provê um resumo de vários estudos relatando resultados de análise de agrupamentos. Comparações entre métodos foram feitas, por exemplo em Popchev e Peneva (1988), que usaram diferentes critérios de similaridade e agregação de dados, e Mandel e Chernyl (1988) que analisaram experimentalmente métodos hierárquicos e o método *k-means* através de simulações de Monte Carlo. Comparações entre métodos convencionais e redes neurais artificiais foram apresentados, por exemplo, em Balakrishnan et al. (1994, 1996). Comparações são em geral muito difíceis e como veremos adiante, cada método impõe uma geometria aos dados, ou à matriz de dissimilaridades (no caso hierárquico), e o sucesso de um método em geral está ligado à aderência ou à coincidência da estrutura do conjunto de dados esta geometria.

Devido aos vários problemas de escolha de parâmetros nos mais variados métodos, em geral os autores recomendam que sejam simuladas várias vezes um determinado método sob condições iniciais diferentes. A constância dos resultados pode evidenciar uma boa escolha do método e dos parâmetros. Porém, a análise dos vários resultados pode ser extremamente trabalhosa. Métodos inteligentes podem ser, no futuro, aplicados para avaliar automaticamente os melhores resultados. Poucos autores dedicaram-se a esta tarefa, por exemplo, Pinkowski (1989) que desenvolveu um sistema especialista que toma decisões baseado em conhecimentos gerados em simulações.

Aplicações em engenharia industrial incluem a tecnologia de grupos, que objetiva maximizar a produção e eficiência concentrando máquinas e mão-de-obra dedicadas a produtos semelhantes em locais próximos. Hausknecht (1988) analisou tipos de automação flexível por métodos de CA havendo encontrado seis grandes grupos de técnicas, o que permitiu determinar características comuns e comparações entre sistemas. De acordo com o autor, a interpretação das características de cada método oferece a possibilidade de geração de regras de configurações de sistemas que podem ser usadas no planejamento e administração de sistemas de produção industrial. Uma abordagem usando lógica nebulosa em tecnologia de grupos foi apresentado em Ben-Arieh e Triantaphyllou (1992).

Na área de computação, o problema de organização e recuperação de informações em grandes bases de dados foi abordada usando métodos de CA, principalmente as técnicas hierárquicas. Basicamente objetiva-se aumentar a eficiência em operações de pesquisa e busca de informações armazenando dados relacionados o mais próximo possível de forma a reduzir o número de acessos às memórias secundárias. A busca é efetuada por níveis de agrupamentos, contrastando com a busca tradicional que é feita sobre registros individuais. Boas introduções ao assunto incluem Ramussen (1990) e Ghosh-Roy *et al.* (1998). Técnicas de análise de correspondência e de agrupamentos foram usados em Missaoui e Frasson (1989) para obter conhecimento do uso de bases de dados. As informações como acesso, gravações, etc., de registros podem ser armazenadas em uma base de conhecimento que aliada a um sistema especialista pode otimizar o processo de busca e armazenamento. Ramussen e Willett (1989) usaram um processador vetorial para implementar busca de informações eficientemente via métodos hierárquicos de CA, por exemplo, os métodos das ligações simples e de Ward. Bouguettaya (1996) e Bouguettaya e Le Viet (1998) igualmente usaram técnicas aglomerativas, este último dedicando-se a comparações, por meio de simulações com dados bidimensionais gerados artificialmente, dos métodos de ligações simples, ligações completas e do centróide. Os autores afirmam que com o advento dos bancos de dados orientados a objetos (OODB) o uso de métodos de CA tornaram-se extremamente importantes e que há necessidade de desenvolvimento de novos algoritmos mais eficientes, que possam efetuar agrupamentos *on-line* e serem dinâmicos de forma a se adaptarem a mudanças nos padrões de acessos. Uma das áreas recentes que mais têm usado ferramentas de CA é a mineração de dados (*data mining*), por exemplo na descoberta de distribuições dos dados, valores discrepantes, confirmação de hipóteses e auxílio em tomadas de decisões. Várias aplicações estão em andamento e grandes empresas investem nesta área, principalmente para extrair conhecimento de grandes bases de dados. A idéia em geral é: os dados são fontes de informação na forma bruta, as informações são necessárias para tomadas de decisões e aumento de competitividade e os bancos de dados, que podem eventualmente chegar a milhões de registros, estão entre os principais capitais da empresa. Como transformar milhões de registros em informações estratégicas é um dos objetivos de *data mining*. Esta tese apresenta ferramenta com objetivo de auxiliar na solução de tais problemas (ver capítulos 5-7). Artigos recentes, tratando ainda com abordagens convencionais, incluem Agrawal *et al.* (1998) e Guha *et al.* (1998).

O uso de CA em auxílio a diagnósticos e classificação é comum em medicina e engenharia biomédica, principalmente com a disponibilidade crescente de instrumentação eletrônica dedicada às várias sub-áreas da medicina. Por exemplo, Cagnoni et al. (1991) abordam o monitoramento da pressão arterial de pacientes cujos dados foram registrados em períodos de 24 horas. CA foi aplicado a um conjunto de parâmetros derivado dos componentes principais da série temporal armazenada. Em outro exemplo, Bortolan et al. (1992)

descrevem o uso da combinação de CA com redes neurais para classificação e diagnóstico de eletrocardiogramas (ECG). Abordagens recentes associadas ao uso de CA em ECG foram apresentadas em Lund *et al.* (1998), onde os autores usaram o mapa auto-organizável de Kohonen, ver capítulo 3, para agrupar formas de onda de ECG e detectar correlações entre variáveis envolvidas. Nevo *et al.* (1991) descrevem um modelo de sistema especialista para análise de dados de anestesia no qual CA é utilizada para detectar novas classes de padrões ou ajustar classes já existentes, objetivando uma classificação eficiente de doenças. Dados de pacientes em UTI também foram analisados por métodos de CA. Avanzolini et al. (1991) usaram um conjunto de 13 variáveis sobre 200 pacientes, obtidas no período de 6 horas após cirurgias cardíacas. CA foi usada para identificar padrões de riscos de falecimento. Os autores identificaram que os padrões de risco elevado e baixo são bastante distintos no espaço 13-dimensional, o que permitiu a construção de regras para sistemas de classificação (análise discriminante) para diagnóstico em novos pacientes.

Estas e muitas outras áreas na ciência ou na indústria têm sido beneficiadas com o uso de técnicas de CA, incluindo biologia e zoologia, geralmente usando métodos hierárquicos (Everitt, 1993); geologia, geografia, e meteorologia sensoriamento remoto; em análises de imagens; marketing e pesquisa de mercado, onde busca-se determinar mercados alvos para produtos e estratégias para maximização de aceitação; pesquisa operacional; engenharia de petróleo; engenharia elétrica, por exemplo em análises de projetos de circuitos, e em processamento de sinais e de imagens, onde métodos de CA e quantização são aplicados em compressão e transmissão de imagens e dados; antropologia, sociologia e criminologia, onde padrões de comportamento são analisados sob diversas variáveis das sociedades; etc.

2.3 Conceitos de distâncias, similaridades e dissimilaridades

Na maioria dos métodos de análise de dados multivariada, a noção de distância é central. A similaridade (ou proximidade) entre dados ou objetos deve ser definida matematicamente. Quase todas as técnicas de agrupamentos envolvem processos de medidas, tanto da magnitude da distância entre dois objetos quanto da magnitude das distâncias entre agrupamentos. Devido ao fato dos dados poderem estar em várias formas, vários critérios de similaridade foram propostos. Em geral, assume-se que os objetos são pontos em um espaço métrico p-dimensional, no qual podemos definir um critério de distâncias.

Seja $\boldsymbol{X}$ um conjunto finito ou infinito de elementos, no qual fazem parte todos os vetores descritos pelas linhas da matriz de dados (por exemplo, como mostrado na tabela 2.1). Seja $\Re$ o conjunto de números reais. Em geral, funções de distâncias entre dois vetores $\boldsymbol{x}$ e $\boldsymbol{y}$

apresentam algumas propriedades básicas. O mapeamento d: $\boldsymbol{X} \times \boldsymbol{X} \rightarrow \Re$ (o qual atribui um número real a cada par de elementos de $\boldsymbol{X}$) é chamado função de distância se, para vetores arbitrários $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{X}$, as condições 2.2-2.4 são válidas[4].

1) A distância $d(\boldsymbol{x}, \boldsymbol{y})$ entre dois vetores é maior que zero, a menos que os vetores sejam idênticos.

$$d(\boldsymbol{x}, \boldsymbol{y}) \geq 0 \qquad (d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y}) \tag{2.2}$$

Em alguns casos consideram-se $d(\boldsymbol{x}, \boldsymbol{x}) = d_0$, e nestes casos, $d(\boldsymbol{x}, \boldsymbol{y}) \geq d_0$, onde d_0 é um número finito real. A relação se torna mínima quando o par de elementos é idêntico.

2) A distância entre dois vetores é simétrica.

$$d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x}) \tag{2.3}$$

3) A distância entre os vetores $\boldsymbol{x}$ e $\boldsymbol{y}$ é menor ou igual à soma das distâncias a um vetor intermediário $\boldsymbol{w}$ ($\boldsymbol{x}, \boldsymbol{y}$ e $\boldsymbol{w} \in \boldsymbol{X}$)

$$d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{w}) + d(\boldsymbol{w}, \boldsymbol{y}) \tag{2.4}$$

A propriedade 3, equação 2.4, corresponde à desigualdade do triângulo da geometria Euclidiana.

Uma função de similaridade s é um mapeamento s: $\boldsymbol{X} \times \boldsymbol{X} \rightarrow \Re$ com as seguintes propriedades:

$$s(\boldsymbol{x}, \boldsymbol{y}) \leq s_0 \qquad (s(\boldsymbol{x}, \boldsymbol{y}) = s_0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y}) \tag{2.5}$$

$$s(\boldsymbol{x}, \boldsymbol{y}) = s(\boldsymbol{y}, \boldsymbol{x}) \tag{2.6}$$

$$\left[s(\boldsymbol{x}, \boldsymbol{y}) + s(\boldsymbol{y}, \boldsymbol{z})\right] \cdot s(\boldsymbol{x}, \boldsymbol{z}) \geq s(\boldsymbol{x}, \boldsymbol{y}) \cdot s(\boldsymbol{y}, \boldsymbol{z}) \tag{2.7}$$

Onde s_0 é um número real. A diferença básica entre d e s reside nas condições 2.2 e 2.5. Se a condição 2.7 for válida, s é chamada uma métrica. A máxima similaridade ocorre quando

[4] Em geral os algoritmos de agrupamentos usam dois tipos diferentes de formato de dados: a matriz de dados X, de tamanho $n \times p$, ou uma matriz de dissimilaridades D_{ij}, de tamanho $n \times n$. Estes dois casos correspondem aos formatos *two-way two-mode data* e *two-way one-mode data*, respectivamente.

objetos são idênticos (condição 2.5). Nesta seção consideraremos, brevemente, apenas funções de distâncias e de similaridades. Um conjunto abrangente, incluindo similaridades para variáveis nominais e ordinais pode ser encontrado em (Anderberg 1973).

Representando os vetores $\boldsymbol{x}$ e $\boldsymbol{y}$ por suas coordenadas, $(x_1, x_2, \ldots x_p)^T$ e $(y_1, y_2, \ldots y_p)^T$, os quais serão usados para corresponder a dois objetos descritos por linhas da matriz de dados, podemos derivar várias medidas de similaridade (ou dissimilaridade). Nem todas respeitam as três propriedades básicas (1-3) descritas anteriormente.

O critério de distância mais empregado na prática é a distância Euclidiana, correspondendo à generalização da distância entre dois pontos em um plano. É derivada da norma L_2 de um vetor $\boldsymbol{x}$. De $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{k=1}^{p} x_k^2} = \sqrt{\boldsymbol{x}^T \boldsymbol{x}}$, obtemos

$$d_2(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2 = \sqrt{(\boldsymbol{x} - \boldsymbol{y})^T (\boldsymbol{x} - \boldsymbol{y})} \tag{2.8}$$

Explicitando todos os elementos dos vetores e incluindo fatores de ponderação, temos:

$$d_e^w(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2^w = \sqrt{w_1 \cdot (x_1 - y_1)^2 + w_2 \cdot (x_2 - y_2)^2 + \cdots + w_p \cdot (x_p - y_p)^2} \tag{2.9}$$

onde w_1, w_2, ..., w_p, são pesos que enfatizam a importância das variáveis no cálculo da distância. No caso convencional, temos $w_i = 1$, $i = 1, 2, \ldots, p$.

A métrica Euclidiana tem, conjuntamente com a norma L_2, a propriedade de que todos os seus valores são invariantes em relação a mapeamentos ortogonais (rotações) dos vetores os quais são descritos por matrizes $Q_{m \times m}$ tal que $Q^T Q = I$ (onde I é a matriz de identidade) (Späth, 1980, 1985). Desta forma temos:

$$\|Q\boldsymbol{x}\|_2^w = \|\boldsymbol{x}\|_2^w \tag{2.10}$$

e

$$d_2^w(Q\boldsymbol{x}, Q\boldsymbol{y}) = d_2^w(\boldsymbol{x}, \boldsymbol{y}) \tag{2.11}$$

Estes mapeamentos ortogonais não são os únicos mapeamentos para os quais d_2^w é invariante. Como um outro exemplo temos as translações ($\boldsymbol{x} \rightarrow \boldsymbol{x} + a$). Segundo Mascarenhas e Velasco (1989), a distância Euclidiana trata-se de uma medida que é invariante a translações, porém não é invariante a transformações lineares em geral. Outras

propriedades da distância Euclidiana foram apresentadas em Anderberg (1973) e Mardia *et al.* (1979).

O problema em se usar distância Euclidiana em agrupamentos é porque tal critério assume covariâncias iguais entre as classes (σ^2), e além disto, $\Sigma = \sigma^2 \cdot I$, onde I é a matriz identidade.

A métrica Euclidiana é um caso especial da métrica de Minkowski, ou norma L_λ

$$\|\boldsymbol{x}\|_\lambda = \left[\sqrt[\lambda]{\sum_{k=1}^{p} |x_k|^\lambda} \right] \qquad \lambda \geq 1. \tag{2.12}$$

Da qual podemos obter, por analogia,

$$d_\lambda(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_\lambda \tag{2.13}$$

A norma L_λ é invariante a translações. A desigualdade $d_m(\boldsymbol{x}, \boldsymbol{y}) \leq d_n(\boldsymbol{x}, \boldsymbol{y})$ vale para todos $\boldsymbol{x}$, $\boldsymbol{y}$ se e somente se $m \geq n$. (Duran & Odell, 1974).

Outros dois casos especiais são quando $\lambda = 1$ e $\lambda = \infty$. As normas

$$\|\boldsymbol{x}\|_1 = \sum_{k=1}^{p} |x_k| \quad e \quad \|\boldsymbol{x}\|_\infty = \max_k |x_k| \tag{2.14}$$

correspondem às métricas

$$d_1(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_1 \qquad \text{e} \qquad d_\infty(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_\infty \tag{2.15}$$

A primeira define a distância de Hamming (eq. 2.16) e a segunda a distância do máximo valor (ou de Chebyshev) (eq. 2.17)

$$d_1(\boldsymbol{x}, \boldsymbol{y}) = d_h(\boldsymbol{x}, \boldsymbol{y}) = |(x_1 - y_1)| + |(x_2 - y_2)| + \cdots + |(x_m - y_m)| \tag{2.16}$$

$$d_\infty(\boldsymbol{x}, \boldsymbol{y}) = \max\left(|(x_1 - y_1)|, |(x_2 - y_2)|, \cdots, |(x_m - y_m)| \right) \tag{2.17}$$

A distância de Hamming é bastante usada em engenharia pelo fato da simplicidade de implementação em hardware. Note que a distância do máximo valor (eq. 2.17) equivale à maior distância entre as coordenadas dos vetores $\boldsymbol{x}$ e $\boldsymbol{y}$.

A outra forma de generalização da distância Euclidiana é obtida definindo

$$\|\boldsymbol{x}\|_B = \sqrt{\boldsymbol{x}^T B\, \boldsymbol{x}} \tag{2.18}$$

no lugar da norma L_2. Aqui, B é uma matriz definida positiva (i.e., uma matriz simétrica tal que $\boldsymbol{x}^T B\, \boldsymbol{x} \geq 0$ para todo $\boldsymbol{x}$, e $\boldsymbol{x}^T B\, \boldsymbol{x} = 0$ se e somente se $\boldsymbol{x} = 0$.

A métrica correspondente é

$$d_B(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(\boldsymbol{x} - \boldsymbol{y})^T B(\boldsymbol{x} - \boldsymbol{y})} \tag{2.19}$$

Em casos simples, B é uma matriz diagonal, os elementos da diagonal são pesos positivos para os componentes dos vetores os quais correspondem a variáveis da matriz de dados. Note que a equação (2.9) é um caso especial da equação (2.19). Mapeamentos ortogonais que fazem a L_2-norma invariante correspondem aqui a matrizes P para as quais $P^T BP = B$, fazendo (2.18) e (2.19) invariantes à translação. Escolhendo adequadamente B, obtemos a métrica de Mahalanobis para objetos, a qual possui propriedades de invariância mais gerais, por exemplo a de ser invariante a qualquer transformação não singular C.

Para fazer isto, precisamos inicialmente da matriz de covariância das variáveis (colunas) de $\boldsymbol{X}$. Denotando a coluna k e a linha i da matriz de dados por $x_{\cdot k}$ e $x_{i\cdot}$ respectivamente; as médias correspondentes são denotadas por $\bar{x}_{\cdot k}$ e $\bar{x}_{i\cdot}$. A matriz de covariância $S = (s_{kj})$ das variáveis é definida por

$$s_{kj} = \frac{1}{n} \sum_{i=1}^{n} (x_{ik} - \bar{x}_{\cdot k})(x_{ij} - \bar{x}_{\cdot j}) \tag{2.20}$$

onde $k, j = 1 \ldots p$. A matriz de covariância dos objetos é definida, de forma análoga, como

$$t_{ij} = \frac{1}{p} \sum_{i=1}^{p} (x_{ik} - \bar{x}_{i\cdot})(x_{jk} - \bar{x}_{j\cdot}) \tag{2.21}$$

onde $i, j = 1, \ldots n$. As matrizes de correlação correspondentes, que são necessárias, entre outras coisas, para métodos de análise fatorial (R- e Q-técnicas), são dadas por

$$r_{kj} = \frac{s_{kj}}{\sqrt{s_{kk} \cdot s_{jj}}} \qquad (k, j = 1, \ldots p) \tag{2.22}$$

e

$$\rho_{ij} = \frac{t_{ij}}{\sqrt{t_{ii} \cdot t_{jj}}} \qquad (i, j = 1, \ldots n) \tag{2.23}$$

respectivamente.

Se escrevermos

$$\tilde{X} = (\tilde{x}_{ik}) = (x_{ik} - \bar{x}_{.k}) \tag{2.24}$$

A matriz S pode ser dada por

$$S = \frac{1}{n} \tilde{X} \cdot \tilde{X}^T \tag{2.25}$$

da qual pode ser visto que, quando as colunas de $\tilde{X}$ são linearmente independentes – como pode ser geralmente assumido quando $n >> p$, a matriz de covariância é definida positiva. Consequentemente S é não singular, e existe uma matriz inversa S^{-1} definida positiva.

Seguindo (Bock, 1974) a distância de Mahalanobis entre dois objetos da matriz de dados é dada por

$$d_S(x_{i.}^T, x_{j.}^T) = \sqrt{(x_{i.} - x_{j.}) S^{-1} (x_{i.} - x_{j.})^T} \tag{2.26}$$

Propriedades desta distância foram discutidas em várias referências, incluindo Mardia et al. (1979). Morrison (1967) recomenda também que sejam dados pesos positivos às variáveis de qualquer magnitude, escolhidas pelo usuário, introduzindo a matriz D em (2.26)

$$d_S(x_{i.}^T, x_{j.}^T) = \sqrt{(x_{i.} - x_{j.}) D S^{-1} (x_{i.} - x_{j.})^T} \tag{2.27}$$

Em aplicações de agrupamentos, a métrica de Mahalanobis sofre a desvantagem de que a matriz é baseada em todos os objetos juntos e não separadamente sobre os objetos em cada grupo, que são desconhecidos (Spath, 1980, p.20). Ainda, o custo computacional necessário

para seu cálculo é muito maior que para outras métricas. Por estas razões, em geral prefere-se usar a distância Euclidiana.

Por outro lado, se os objetos forem normalizados, isto é,

$$\| x_{i\cdot}^T \|_2 = 1 \quad (i = 1,..., p) \tag{2.28}$$

S degenera para a matriz identidade, implicando que, após esta transformação, a métrica de Mahalanobis fica igual à métrica Euclidiana.

A normalização mencionada tem ainda um outro efeito na métrica Euclidiana (lei do coseno):

$$\begin{aligned} d_2(x_{i\cdot}^T, x_{j\cdot}^T)^2 &= \|x_{i\cdot}^T - x_{j\cdot}^T\|_2^2 = \|x_{i\cdot}^T\|_2^2 + \|x_{j\cdot}^T\|_2^2 - 2x_{i\cdot}x_{j\cdot}^T \\ &= 2(1-\rho_{ij}), \end{aligned} \tag{2.29}$$

onde ρ_{ij} é a matriz de correlação. A igualdade 2.29 pode ser vista como um relacionamento entre funções de distância e similaridade.

Entre as várias medidas de similaridade existentes, inclui-se o coseno do ângulo ϕ entre dois vetores $\boldsymbol{x}$ e $\boldsymbol{y}$, definido por:

$$\cos\phi = \frac{\boldsymbol{x}^T \boldsymbol{y}}{\|\boldsymbol{x}\| \cdot \|\boldsymbol{y}\|} \tag{2.30}$$

Esta medida é invariante frente à rotação e à mudança de escala, porém não é invariante frente à translação e à transformações lineares em geral.

Para vetores binários, por exemplo, temos a medida de Tanimoto

$$T(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x}^T \boldsymbol{y}}{\boldsymbol{x}^T \boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{y} - \boldsymbol{x}^T \boldsymbol{y}} \tag{2.31}$$

Esta medida (Mascarenhas e Velasco, 1989) pode ser interpretada como a razão entre o número de atributos comuns a $\boldsymbol{x}$ e $\boldsymbol{y}$ e o número de atributos possuídos por $\boldsymbol{x}$ ou $\boldsymbol{y}$ (definindo que $\boldsymbol{x}$ possui o k-ésimo atributo se o k-ésimo componente de $\boldsymbol{x}$ é 1).

2.4 Métodos Hierárquicos

Desenvolvidos inicialmente no campo da biologia (Sokal & Snead, 1973), as técnicas hierárquicas ganharam popularidade devido a vários fatores, como versatilidade, simplicidade e variedade de métodos disponíveis, como também à noção intuitiva de que graus relativos de semelhança entre os objetos poderiam ser visualizados em uma representação hierárquica, como por exemplo, uma árvore.

Métodos hierárquicos produzem uma sucessão de partições, cada qual correspondendo a um diferente número de agrupamentos. Tais métodos são em geral subdivididos em técnicas aglomerativas e divisórias. Os métodos aglomerativos consideram no início que os n objetos são n subgrupos e por meio de uniões sucessivas, uma de cada vez, chega-se a um único agrupamento contendo todos os objetos no final do processo. Os métodos divisórios, por outro lado, consideram inicialmente um agrupamento de n objetos, e por sucessivas divisões chega-se a n subgrupos, cada um contendo um único objeto. Os métodos hierárquicos produzem seqüências aninhadas de partições do conjunto de dados. O resultado de uma classificação hierárquica geralmente é representado por meio de um dendrograma que ilustra as fusões (ou divisões) feitas em cada estágio sucessivo da análise. A seguir, descrevemos brevemente alguns métodos aglomerativos. Nesta tese não consideraremos métodos divisórios por estes serem considerados bastante ineficientes e por esta razão, serem menos expressivos que as técnicas aglomerativas.

2.4.1 Métodos Aglomerativos

Mais comuns dentre os métodos hierárquicos, as técnicas aglomerativas operam, geralmente, sobre uma matriz de similaridades ou dissimilaridades, D_{ij} $(i, j = 1, 2, \ldots n)$[5], produzindo uma seqüência de partições dos dados, P^n, P^{n-1}, ..., P^1. A primeira, P^n, consiste de n agrupamentos contendo um elemento apenas, e a última, P^1, consiste de um agrupamento contendo todos os n objetos (Everitt, 1993). O que basicamente diferencia os métodos é a escolha da ultra-métrica[6] a ser adotada, ou o critério de união entre os agrupamentos durante as sucessivas fusões, e como se recalculam as distâncias entre um agrupamento formado a todos os outros restantes. O procedimento geral pode ser descrito em poucos passos:

[5] Pelo fato da matriz D_{ij} ser em geral simétrica e com diagonal nula costuma-se armazenar apenas $n.(n-1)/2$ valores de similaridade (ou dissimilaridade).

1. Início: Cada agrupamento C_1, C_2, ..., C_n contém um único objeto.
2. Determine o par de agrupamentos distintos (C_i, C_j), $i \neq j$, com maior (menor) grau de similaridade (dissimilaridade).
3. Forma-se um novo agrupamento pela união dos agrupamentos C_i e C_j, i.e., $C_k = C_i \cup C_j$. Calcula-se as novas medidas de similaridade (dissimilaridade) entre o novo agrupamento C_k e todos os outros restantes (D_{kl}). Diminui-se o número total de agrupamentos em 1.
4. Os passos 2 e 3 são executados (n-1) vezes, até que todos os objetos estejam em um único agrupamento.

Em relação ao passo 4, obviamente pode-se desejar parar a fusão dos agrupamentos em um determinado nível, w, do dendrograma, sem que todos os objetos estejam totalmente agrupados, reduzindo o número total de passos.

Lance & Williams (1967) desenvolveram uma fórmula de recorrência generalizada que permite a determinação das novas distâncias entre o agrupamento formado (C_k) e todos os outros l agrupamentos, D_{kl}, onde $C_k = C_i \cup C_j$ e C_l é um outro agrupamento qualquer. Tal fórmula, apresentada na equação 2.32, tem a vantagem de necessitar, em cada estágio da análise, apenas das informações da matriz de similaridades (ou dissimilaridades) do estágio anterior e funciona para muitos dos métodos aglomerativos.

$$D_{kl} = \alpha_i \cdot D_{ki} + \alpha_j \cdot D_{jk} + \beta \cdot D_{ij} + \gamma \cdot | D_{ik} - D_{jk} | \qquad (2.32)$$

onde os parâmetros α_i, α_j, β e γ definem cada um dos métodos. A seguir comentamos brevemente alguns dos métodos mais utilizados e os seus respectivos valores dos parâmetros em (2.32). Por razões de simplicidade, consideraremos daqui em diante apenas a palavra dissimilaridade para representar as relações entre os objetos, onde a menor dissimilaridade representará o maior grau de semelhança.

2.4.1.1 Ligação simples (LS)

Também denominado de método dos vizinhos mais próximos, sendo o mais simples dentre os métodos aglomerativos, é caracterizado por considerar a dissimilaridade entre dois agrupamentos, C_i e C_k, como a menor dissimilaridade dentro de cada par de objetos

[6] Métrica utilizada para fusão de agrupamentos.

formados por um objeto pertencente a C_i e outro pertencente a C_k (Florek et al., 1951; Sneath, 1957; Johnson, 1967).

$$D_{i\cup k,j} = \min\{D_{i,j}, D_{k,j}\}$$

para todos os objetos $j \neq i, k$.

Os coeficientes da equação 2.32 assumem valores $\alpha_i = \alpha_j = 1/2$, $\beta = 0$ e $\gamma = -1/2$. Apesar de bastante estudado, e largamente usado, este método apresenta o 'efeito de cadeia', o qual possibilita a descoberta de agrupamentos alongados no espaço, mas tem a desvantagem de conectar agrupamentos que idealmente deveriam estar separados. Isto pode ocorrer quando existirem alguns pontos ligando estes dois agrupamentos.

2.4.1.2 Ligação completa (LC)

Neste caso, a dissimilaridade entre dois agrupamentos, C_i e C_k, é definida como a maior das dissimilaridades dentro de cada par de objetos formados por um objeto pertencente a C_i e outro pertencente a C_k, isto é, para todos os objetos $j \neq i, k$:

$$D_{i\cup k,j} = \max\{D_{i,j}, D_{k,j}\}$$

Os coeficientes da equação 2.32 assumem valores $\alpha_i = \alpha_j = 1/2$, $\beta = 0$ e $\gamma = 1/2$. Verifica-se que o par de elementos mais afastados entre os agrupamentos $C_{i\cup k}$ e o C_j serão usados no cálculo da dissimilaridade resultante. A tendência deste algoritmo é formar vários agrupamentos de tamanhos pequenos e compactos, i.e., com grande homogeneidade (Kopp, 1978). Pode ocorrer que objetos relativamente similares permaneçam em agrupamentos diferentes em boa parte da análise, sendo unidos somente nos estágios finais (Kaufman & Rousseeuw, 1990). Um outro inconveniente é a sensibilidade à presença de valores discrepantes (*outliers*).

2.4.1.3 Outros métodos

Outros métodos importantes incluem: média das ligações, McQuitty, mediana das ligações, centróide e o método de Ward. A tabela 2.2 apresenta os valores dos coeficientes da fórmula de recorrência de Lance e Williams (1967) para cada método. Uma tabela e uma fórmula ainda mais completas foram apresentadas em Jambu (1978).

TABELA 2.2: ESPECIFICAÇÕES DE SETE MÉTODOS DE AGRUPAMENTOS HIERÁRQUICOS

Método	α_i	β	γ
Ligação simples (LS)	0.5	0	-0.5
Ligação completa (LC)	0.5	0	0.5
Média de ligações (ML)	$\frac{\lvert i\rvert}{\lvert i\rvert+\lvert j\rvert}$	0	0
McQuitty	0.5	0	0
Mediana das ligações (MdL)	0.5	-0.25	0
Centróide	$\frac{\lvert i\rvert}{\lvert i\rvert+\lvert j\rvert}$	$-\frac{\lvert i\rvert\cdot\lvert j\rvert}{(\lvert i\rvert+\lvert j\rvert)^2}$	0
Ward	$\frac{\lvert i\rvert+\lvert k\rvert}{\lvert i\rvert+\lvert j\rvert+\lvert k\rvert}$	$-\frac{\lvert k\rvert}{\lvert i\rvert+\lvert j\rvert+\lvert k\rvert}$	0

(*) Obs.: $\lvert i\rvert$ é a cardinalidade do agrupamento i, i.e., número de objetos.

O método da média das ligações (ML) situa-se entre o método da ligação simples e o da ligação completa, usando uma média das dissimilaridades entre todos os pares de objetos, com cada par formado por um objeto de cada agrupamento envolvido. O agrupamento é caracterizado pela média de todas as dissimilaridades entre os seus membros, sendo menos sensível a *outliers* do que o LC e gerando agrupamentos mais homogêneos do que o LS. De acordo com Kaufman & Rousseeuw (1990), apesar da tendência de determinar agrupamentos com formas esféricas, o ML é relativamente robusto para lidar com grupos de outras formas.

O método McQuitty, relativamente pouco discutido na literatura, não define explicitamente a dissimilaridade entre os agrupamentos, sendo definida apenas a equação recursiva para determinar as dissimilaridades entre os agrupamentos formados e os grupos restantes. Kaufman & Rousseeuw (1990) descrevem que resultados contraditórios podem ocorrer no emprego deste método, por exemplo, quando um objeto possui a mesma dissimilaridade a outros dois objetos, a ordem da união pode alterar o resultado final.

Alguns métodos, como por exemplo ML e Centróide, só possuirão uma interpretação mais clara quando aplicados em problemas cuja matriz de dissimilaridades esteja no espaço Euclidiano. No caso do centróide, os agrupamentos são representados pelo centróide, ou centro de massa, e em cada estágio do processo, unem-se os agrupamentos que possuam menor distância entre seus centróides. Sendo $\bar{x}_k$ o centróide do agrupamento k, um vetor

no espaço p-dimensional de atributos, $\bar{x}_k = (\bar{x}_{k1}, \bar{x}_{k2}, \ldots, \bar{x}_{kp})$, e onde $\bar{x}_{kf} = \frac{1}{|k|} \sum_{g \in X^k} (x_{gf})$, onde $|k|$ é a cardinalidade do agrupamento C_k, as coordenadas do novo agrupamento C_r formado pela união dos agrupamentos k e l, são dadas por

$$\bar{x}_r = \frac{|k| \cdot \bar{x}_k + |l| \cdot \bar{x}_l}{|k| + |l|} \tag{2.33}$$

e a dissimilaridade entre os centróides $\bar{x}_k$ e $\bar{x}_l$ é dada pela distância Euclidiana, $\|\bar{x}_k - \bar{x}_l\|^2$. Vemos que a cardinalidade dos agrupamentos influi no centróide do agrupamento resultante ponderando a média entre os centróides $\bar{x}_k$ e $\bar{x}_l$, atraindo-o assim para o agrupamento que possuir maior cardinalidade. A tendência deste método é formar um ou dois agrupamentos grandes, contendo a maioria dos objetos.

No caso da mediana das ligações (MdL), também denominado método de Gower (Gower, 1967), a influência da cardinalidade dos agrupamentos não é levada em consideração na formação do novo centro do agrupamento resultante. Os agrupamentos são representados, neste caso, não pelo centro de massa de seus elementos componentes, mas pelo valor mediano, o qual denominaremos simplesmente *centro*. De forma similar ao método do centróide, a dissimilaridade entre dois agrupamentos é dada pela distância Euclidiana entre seus centros, porém as coordenadas do novo centro são obtidas fazendo $|k| = |l| = 1$ na equação (2.33), i.e., $\bar{x}_r = \frac{1}{2}(\bar{x}_k + \bar{x}_l)$. Devido ao fato de que os centros dos agrupamentos não são definidos de forma única, a fórmula recorrente de Lance e Williams contendo os coeficientes para MdL não explicita uma definição clara para a dissimilaridade entre os agrupamentos, sendo dependente da ordem na qual os grupos são formados, o que também pode levar a resultados contraditórios. Na prática os resultados são, em certo grau, semelhantes ao método do centróide, e a escolha entre tais métodos depende de objetivos práticos, principalmente quando existe alguma informação *a priori*, como por exemplo a existência ou não de agrupamentos pequenos porém importantes.

Finalmente, o método Ward propõe que os agrupamentos sejam formados objetivando otimizar um critério (Ward, 1963). A fusão entre agrupamentos em geral prioriza a união dos grupos que minimiza a variância, ou a soma dos quadrados dos desvios (ou distâncias) S_w em relação à média dentro dos grupos. Para um dado agrupamento X^r definimos $S_w(X^r)$ como

$$S_w(X^r) = \sum_{i \in X^r} \|x_i - \bar{x}_r\|^2 . \tag{2.34}$$

A cada estágio considera-se a S_w total entre os agrupamentos, S_{wT}, dada por $S_{wT} = \sum_{i=1}^{K} S_w(\boldsymbol{X}^i)$, onde K é o número total de grupos no estágio em questão. Inicialmente, cada agrupamento é composto de um único elemento, e $S_{wT} = 0$. Cada união de grupos, $\boldsymbol{X}^r = \boldsymbol{X}^k \bigcup \boldsymbol{X}^l$, acrescenta $\Delta S_{wT} = S_w(\boldsymbol{X}^r) - S_w(\boldsymbol{X}^k) - S_w(\boldsymbol{X}^l)$ à variância total, S_{wT}. O objetivo, em cada estágio, é unir dois grupos nos quais tal acréscimo ΔS_{wT} seja mínimo. A dissimilaridade entre os grupos $\boldsymbol{X}^k$ e $\boldsymbol{X}^l$ é dada por

$$D_{kl} = \frac{|k| \cdot |l|}{|k| + |l|} \left\| \bar{\boldsymbol{x}}_k - \bar{\boldsymbol{x}}_l \right\|^2 \tag{2.35}$$

onde $|k|$ e $|l|$ representam as cardinalidades dos grupos k e l, respectivamente. O centro do novo agrupamento $\boldsymbol{X}^r$ é dado por $\bar{\boldsymbol{x}}_r = \frac{|k| \cdot \bar{\boldsymbol{x}}_k + |l| \cdot \bar{\boldsymbol{x}}_l}{|k| + |l|}$. O método de Ward apresenta uma tendência de formar agrupamentos com o mesmo número de objetos. Da equação (2.35) verifica-se que, se um agrupamento está situado à mesma distância em relação a outros dois agrupamentos com diferentes números de objetos, o método priorizará a união com o agrupamento com menor número de objetos. Apesar de ser um dos métodos mais usados na prática, problemas como a sensibilidade a *outliers* podem afetar seu desempenho.

Vários outros métodos foram propostos na literatura, como por exemplo o método *beta-flexível* (Lance & Williams, 1967; Milligan & Cooper, 1985), e o método k-ésimo vizinho mais próximo (Wong & Lane, 1983).

As desvantagens principais de métodos hierárquicos são: (*i*) fusão de agrupamentos em um determinado estágio não pode ser corrigido em estágios subsequentes; (*ii*) em geral, eles requerem espaço de memória de ordem $O(N^2)$ e tempo de processamento $O(N^3)$, onde N é o número de registros (cardinalidade) do conjunto de dados; (*iii*) os resultados podem ser difíceis de interpretar, especialmente para conjuntos de dados grandes; (*iv*) para se determinar onde cortar o dendrograma (qual o número de agrupamentos ideal), ou onde parar o método, há necessidade de um segundo critério (Miligan & Cooper, 1985).

2.5 Métodos Particionais

Diferentemente dos métodos hierárquicos, os métodos particionais produzem uma partição dos n objetos em K agrupamentos, geralmente otimizando uma função objetivo. Existem várias formas de escolher critérios para efetuar a partição, assim como algoritmos de otimização que podem ser usados em conjunto. Vantagens em relação às técnicas hierárquicas incluem a possibilidade de mudanças de pertinência de objetos em relação a um agrupamento durante todo o processo de formação dos agrupamentos, e possibilidade de trabalhar com bases de dados maiores. Em aplicações de mineração de dados (*data mining*) pode-se ter bases de dados bastante extensas, ex. 10 milhões de registros (Michaud, 1997). Os métodos particionais em geral requerem espaço e tempo de ordem O(N), onde N é o número de registros no conjunto de dados.

Uma desvantagem inerente a estes métodos é que, em geral, as funções objetivo usadas partem da premissa de que o número de agrupamentos, K, é conhecido *a priori*. Na hipótese de termos escolhido um valor K' inadequado o método irá impor, pelo uso de técnicas de otimização, K' agrupamentos aos dados. Um outro problema é que tais métodos de otimização em geral são sensíveis às condições iniciais (a escolha dos K protótipos iniciais), podendo gerar partições diferentes, quando são feitas várias simulações, para um mesmo conjunto de dados.

Durante o processo de otimização pela busca da solução ideal, cada partição produz um valor *f(n, k)* permitindo comparações entre diferentes partições obtidas. A maioria das funções objetivo usadas em CA têm relação com as seguintes matrizes calculadas a partir de uma partição dos dados:

$$\mathbf{T} = \frac{1}{n}\sum_{i=1}^{K}\sum_{j=1}^{n_i}\left(x_{ij} - \bar{x}\right)\cdot\left(x_{ij} - \bar{x}\right)^T \tag{2.36}$$

$$\mathbf{W} = \frac{1}{n-K}\sum_{i=1}^{K}\sum_{j=1}^{n_i}\left(x_{ij} - \bar{x}_i\right)\cdot\left(x_{ij} - \bar{x}_i\right)^T \tag{2.37}$$

$$\mathbf{B} = \sum_{i=1}^{K} n_i \cdot \left(\bar{x}_i - \bar{x}\right)\cdot\left(\bar{x}_i - \bar{x}\right)^T \tag{2.38}$$

onde n é o número de objetos no conjunto de dados, n_i é o número de objetos no agrupamento i, K é o número de agrupamentos, $\bar{x}_i$ é o vetor de médias ou centróide do agrupamento i, e $\bar{x}$ é o vetor de média total do conjunto de dados.

Estas matrizes, de dimensão $p \times p$, representam a variância total (**T**), a variância intra-grupo (**W**) e a variância entre-grupos (**B**), e satisfazem (Jain & Dubes, 1988; Everitt, 1993)

$$\mathbf{T} = \mathbf{W} + \mathbf{B}\,.$$

Um critério intuitivo de escolha de partições seria o que minimizassem **W**, i.e., escolhesse uma partição com mínima variância total intra-grupos, para um número fixo de agrupamentos, K. Isto é exatamente o que faz, usando uma heurística, um dos métodos mais populares das técnicas particionais, o *k-means*. Pelo fato de que **T** depende exclusivamente do conjunto de dados, sendo fixo, a minimização de **W** é equivalente à maximização de **B**.

De uma forma mais clara, suponha que o conjunto de n padrões em p dimensões deva ser particionado em K agrupamentos $\{ C_1, C_2, \ldots, C_K \}$, onde cada agrupamento C_k possui n_k objetos e que cada objeto pertença a um único agrupamento. Desta forma

$$\sum_{k=1}^{K} n_k = n$$

e $\bar{x}_k$, o vetor de médias ou centróide do agrupamento k, é dado por

$$\bar{x}_k = \frac{1}{n_k} \sum_{i=1}^{n_k} u_{ik} \cdot x_i$$

onde cada elemento da matriz u, u_{ik}, estabelece a pertinência do padrão x_i ao agrupamento k. Neste caso o padrão x_i pertencerá ao agrupamento C_k que possua o vetor de médias $\bar{x}_k$ mais próxima, i.e., considerando distâncias Euclidiana,

$$u_{ik} = 1 \quad \Leftrightarrow \quad \left\| x_i - \bar{x}_k \right\|^2 < \left\| x_i - \bar{x}_j \right\|^2, \quad k \neq j, \quad j = 1, 2, \ldots K. \tag{2.39}$$

No caso mais simples u_{ik} é uma matriz com elementos binários (0's e 1's) e com tamanho $n \times K$. No caso mais geral, de partições nebulosas, $u_{ik} \in [\, 0, 1]$ e valores intermediários entre 0 e 1 refletem o grau de pertinência de um objeto a um determinado agrupamento.

Consideramos, por enquanto que $u_{ik} = 0$ ou 1, e que $\sum_{k=1}^{K} u_{ik} = 1$ e $\sum_{i=1}^{n} u_{ik} > 0$, ou seja, cada objeto pertencerá exclusivamente a um agrupamento e no mínimo um objeto será atribuído a um dado agrupamento C_k. Desta forma, o número de objetos em um dado agrupamento C_k pode ser expresso por $n_k = \sum_{i=1}^{n} u_{ik}$.

O erro quadrático (Jain & Dubes, 1988) para o agrupamento C_k é a soma das distâncias entre cada padrão $x_i \in C_k$ e sua média $\bar{x}_k$

$$e_k^2 = \sum_{i=1}^{n_k} u_{ik} \cdot (x_i - \bar{x}_k)^T (x_i - \bar{x}_k)$$

Vemos que e_k^2 expressa a variabilidade intra-grupos. O erro quadrático total para os K agrupamentos é obtido pela soma dos K erros quadráticos

$$E_K^2 = \sum_{k=1}^{K} e_k^2 \equiv \sum_{k=1}^{K} \sum_{i=1}^{n_k} u_{ik} \cdot (x_i - \bar{x}_k)^T (x_i - \bar{x}_k)$$

A partição encontrada que minimiza E_K^2 para um número fixo de K agrupamentos é denominada partição de variância mínima. Outra forma seria minimizar a soma das distâncias quadráticas dentro dos agrupamentos (Gordon & Henderson, 1977; Jain & Dubes, 1988).

A minimização de E_K^2 é equivalente à minimização do traço da matriz **W** (ou à maximização do traço da matriz **B**), (Jain & Dubes, 1988). Este critério é usado, implicitamente, em métodos como o *k-means* (MacQueen, 1967) e o *Isodata* (Ball & Hall, 1967). Um outro critério sugerido por Friedman & Rubin (1967) é a maximização do traço da matriz obtida pelo produto da matriz de variância entre-grupos com o inverso da matriz de variâncias intra-grupo, i.e., traço($\mathbf{BW}^{-1}$). A matriz $\mathbf{BW}^{-1}$ também é encontrada em outros contextos, como em análise de variáveis canônicas e em análise multivariada de variância (Everitt, 1993). Outra opção é considerar a minimização do determinante de **W** (Marriott, 1982). O fundamento provêm de análise multivariada da variância onde um dos testes para determinar se dois vetores de médias são iguais é baseado na razão dos determinantes de **T** e **W**. Valores elevados de [*det*(**T**) / *det*(**W**)] indicam que as médias dos vetores diferem. Como **T** independe da partição obtida, a maximização de [*det*(**T**) / *det*(**W**)]

é equivalente à minimização de $det(\mathbf{W})$. Pode-se expressar o traço($\mathbf{BW}^{-1}$) e [$det(\mathbf{T})$ / $det(\mathbf{W})$] em termos dos auto-valores de $\mathbf{BW}^{-1}$, λ_i,

$$traço(\mathbf{BW}^{-1}) = \sum_{i=1}^{p} \lambda_i$$

e

$$\frac{det(\mathbf{T})}{det(\mathbf{W})} = \prod_{i=1}^{p} (1+\lambda_i).$$

A maioria dos critérios citados aplica-se principalmente quando os dados são apresentados na forma contínua, i.e., $x_i \in \Re^p$. Outros critérios mais apropriados a diferentes tipos de dados, como variáveis ordinais ou binárias, são descritos em Späth (1985).

Devido à impraticabilidade de examinar todas as possíveis partições uma vez escolhida a função objetivo (ver seção 2.1), métodos de otimização, semelhantes ao gradiente descendente, são usados na esperança de encontrar uma solução aceitável. No caso do *k-means*, o procedimento geral pode ser descrito em poucos passos:

1. Início: Escolher o número de agrupamentos, K, e os valores iniciais dos K protótipos, ou vetores de média, $\bar{x}_k(0)$, k = 1, 2, ..., K. Outros parâmetros que podem fazer parte da inicialização, relacionados a esquemas de finalização do algoritmo incluem o número máximo de iterações, $t_{máx}$, um valor pequeno ε.

2. Para t = 1, ..., $t_{máx}$, classifique os objetos x_i, i = 1, 2, ..., n, como pertencentes ao agrupamento C_k que satisfaça a equação (2.39), i.e.,

$$u_{ik}(t) = 1 \quad \Leftrightarrow \quad \left\|x_i - \bar{x}_k(t-1)\right\|^2 < \left\|x_i - \bar{x}_j(t-1)\right\|^2, \quad k \neq j, \quad j = 1, 2, \ldots K.$$

3. Determine o valor da função objetivo com a partição obtida, por exemplo, no caso onde estamos minimizando o erro quadrático, $e_k^2(t)$ e $E_K^2(t)$.

4. Recalcule os vetores de médias, $\bar{x}_k(t)$, k = 1, 2, ..., K, baseado na informação $u_{ik}(t)$.

5. Repetir passos 2 a 4 enquanto $t < t_{máx}$, ou $\boldsymbol{u}_{ik}(t) - \boldsymbol{u}_{ik}(t\text{-}1) \neq 0$, ou $\left|\boldsymbol{E}_K^2(t) - \boldsymbol{E}_K^2(t-1)\right| > \varepsilon$.

Obviamente, diversas pequenas modificações podem ser adaptadas ao algoritmo descrito, dependendo de particularidades da implementação. Basicamente, dada uma partição no passo t, capturada na matriz de pertinências $\boldsymbol{u}_{ik}(t)$, recalculam-se os vetores de média dos K agrupamentos, $\bar{\boldsymbol{x}}_k(t)$, $k = 1, 2, \ldots, K$. No passo $t+1$ todos os padrões são novamente classificados de acordo com o novo conjunto de médias, originando a matriz $\boldsymbol{u}_{ik}(t+1)$. O algoritmo pode ser interrompido por várias formas, sendo a mais comum quando a pertinência dos objetos em relação aos agrupamentos se estabiliza, i.e., $\boldsymbol{u}_{ik}(t) - \boldsymbol{u}_{ik}(t\text{-}1) = 0$. Uma outra forma, como descrito no passo 5, seria parar o processo para incrementos no erro quadrático total menores que ε, ou quando extrapola-se o número máximo de iterações, $t_{máx}$. Os vetores de média $\bar{\boldsymbol{x}}_k(t)$, $k = 1, 2, \ldots, K$, migram ao longo das iterações para posições estáveis, e infelizmente, em muitos casos, para mínimos locais de $\boldsymbol{E}_K^2$. O resultado deste método de otimização pode, em muitos casos, ser drasticamente afetado pela escolha das condições iniciais. Entretanto, em bases de dados bem estruturadas, em geral, espera-se a convergência para um mesmo mínimo, global, a partir da maioria das configurações iniciais (Hartigan, 1975). Comportamentos como convergência lenta e resultados de agrupamentos bastante diferentes para diferentes configurações iniciais pode indicar que o número de agrupamentos escolhido, K, esteja errado, ou que os dados não possuam estrutura de agrupamentos (Marriott, 1982).

A escolha correta do número de agrupamentos em um determinado conjunto de dados multidimensional é um do problemas mais fundamentais e não solucionados em CA. Esquemas de validação das partições podem ser usadas (Jain & Dubes, 1988) porém muito da responsabilidade da análise freqüentemente é atribuída ao usuário do método. O problema quando escolhe-se K erroneamente é que o método irá impor uma estrutura aos dados, no lugar de buscar a estrutura inerente a estes.

2.6 Métodos baseados em lógica nebulosa

A teoria de conjuntos nebulosos teve impulso após o trabalho de Zadeh (1965). No contexto de classificação, os métodos baseados em lógica nebulosa (ou *fuzzy*) relaxam a pertinência dos objetos às classes, $\boldsymbol{u}_{ik}$, que neste caso podem assumir qualquer valor real no intervalo [0, 1]. Os conjuntos nebulosos são extensões da teoria clássica dos conjuntos, e têm sido utilizados nas mais variadas aplicações devido a sua flexibilidade em representar

incertezas, como por exemplo no controle de servomecanismos. Os estatísticos geralmente denominam as técnicas de agrupamentos *fuzzy* de técnicas de *clumping*. Apesar dos objetos poderem pertencer a mais de uma classe, geralmente restringe-se a função de pertinência de forma que

$$\sum_{k=1}^{K} u_{ik}(x_i) = 1, \tag{2.40}$$

ou seja, a soma das pertinências de um padrão x_i deve ser igual a 1, onde K é o número escolhido de classes. Em várias áreas de reconhecimento de padrões têm-se usado modelos fuzzy, incluindo modelos de redes neurais. Uma ótima referência do assunto é Bezdek & Pal (1992).

O método *fuzzy k-means* (FKM) gera partições nebulosas do conjunto de dados X minimizando, iterativamente, um funcional ou função objetivo (Bezdek, 1981). Uma partição nebulosa satisfaz a equação (2.40) e também

$$0 < \sum_{i=1}^{n} u_{ik}(x_n) < n \tag{2.41}$$

onde n é o número de objetos na base de dados.

Dado o conjunto de objetos $X = \{x_1, x_2, ..., x_n\}$, onde cada objeto x_k é um vetor no espaço p-dimensional, $x_k = \{x_{k1}, x_{k2}, ..., x_{kp}\} \in \Re^p$, o objetivo é encontrar a pseudo-partição *fuzzy* e os centros dos agrupamentos associados pelo qual a estrutura de dados é representada da melhor forma possível, segundo alguma função objetivo. Para efetuar esta tarefa, devemos ter critérios de como expressar a idéia de associações dos padrões às classes, i.e., ser mais forte dentro das classes, e mais fraco fora destas. Isto pode ser implementado a partir da idéia de *índice de desempenho*, que é baseado nos centros dos agrupamentos. Dada uma pseudo-partição $P = \{ C_1, C_2, ..., C_k\}$, os K centros dos agrupamentos $(v_1, v_2, ..., v_k)$ associados às partições são calculados pela fórmula

$$v_i = \frac{\sum_{i=1}^{n} [u_{ik}(x_i)]^m \cdot x_i}{\sum_{i=1}^{n} [u_{ik}(x_i)]^m} \tag{2.42}$$

onde $m > 1$ é um número real que controla a influência dos graus de pertinência. O vetor v_i, calculado pela equação (2.42), que é o centro do agrupamento da classe *fuzzy* C_i, pode ser visto como uma média ponderada dos dados em C_i. O peso dos dados x_i é a potência m do grau de pertinência de x_i no conjunto *fuzzy* C_i.

O índice de desempenho $J_m(P)$ da pseudo-partição P é definido em termos dos centros dos agrupamentos pela fórmula

$$J_m(P) = \sum_{i=1}^{n}\sum_{k=1}^{K}[u_{ik}(x_i)]^m \cdot \|x_i - v_k\|^2 \qquad (2.43)$$

onde $\|\cdot\|$ é o produto interno no espaço $\Re^p$, e $\|x_k - v_i\|^2$ representa a distância entre x_k e v_i. O índice de desempenho mede a soma das distâncias ponderadas entre os centros dos agrupamentos e dos elementos os agrupamentos *fuzzy*. Quanto menor $J_m(P)$, melhor a pseudo-partição *fuzzy* P. Assim, o objetivo do método FKM é encontrar uma pseudo-partição P que minimiza o índice de desempenho $J_m(P)$, sendo portanto, um problema de otimização. A seguir, descrevemos o algoritmo desenvolvido por J. Bezdek (1981) para resolver o problema de otimização. Assume-se que o número de agrupamentos é conhecido, K, e escolhe-se um critério de distância, como por exemplo, a distância Euclidiana. Dois outros parâmetros escolhidos são o valor $m \in (1, \infty)$ e uma pequena constante ε, que servirá como critério de parada.

Algoritmo *Fuzzy K-Means*:

Passo 1. Faça $t = 0$. Selecione uma pseudo-partição *fuzzy* $P^{(0)}$.

Passo 2. Calcule o centro dos k agrupamentos $(v_1^{(t)}, v_2^{(t)}, ..., v_k^{(t)})$ através da equação 2.42 para $P^{(t)}$ e o valor escolhido de m.

Passo 3. Atualize $P^{(t+1)}$ através de:

Para cada $x_i \in X$, Se $\left\| x_i - v_k^{(t)} \right\|^2 > 0$, então para todo k, $k = 1, 2, \ldots, K$

$$u_{ik}^{(t+1)}(x_i) = \left[\sum_{j=1}^{K} \left(\frac{\left\| x_i - v_k^{(t)} \right\|^2}{\left\| x_i - v_j^{(t)} \right\|^2} \right)^{\frac{1}{m-1}} \right]^{-1}.$$

Caso $\left\| x_i - v_k^{(t)} \right\|^2 = 0$, para algum $k \in K \subseteq \aleph_k$, onde $\aleph_k$, é o conjunto dos números naturais, então defina $u_{ik}^{(t+1)}(x_i)$ como qualquer número real não negativo que satisfaça

$$\sum_{k \in K} u_{ik}^{(t+1)}(x_i) = 1$$

e defina $u_{ik}^{(t+1)}(x_i) = 0$, $\forall\, k \in \aleph_k - K$.

Passo 4. Compare $P^{(t)}$ e $P^{(t+1)}$. Se $\| P^{(t+1)} - P^{(t)} \| \leq \varepsilon$, então pare. Caso contrário, incremente t, e volte ao passo 2.

O parâmetro m é selecionado de forma *ad hoc*, i.e., depende do problema. Quando $m \rightarrow 1$, o *fuzzy k-means* converge para o método *k-means* clássico. Por outro lado, quando $m \rightarrow \infty$, todos os centros dos agrupamentos tendem ao centróide do conjunto de dados X. Isto é, a partição torna-se mais *fuzzy* com o aumento de m. Apesar de não haver uma base teórica para a escolha ótima de m, o algoritmo converge para qualquer $m \in (1, \infty)$. Note que $\| P^{(t+1)} - P^{(t)} \|$ é a distância entre $P^{(t+1)}$ e $P^{(t)}$ no espaço $\Re^{n \times p}$.

A norma do produto interno pode ser alterada para adaptar-se ao formato dos agrupamentos gerados pelo algoritmo FKM: hiper-esférica, hiper-elipsoidal, um sub-espaço linear, etc. O FKM apresenta várias vantagens enquanto gerador de funções de pertinência para processamento posterior: é não supervisionado, pode ser usado com um número qualquer de atributos e de classes, e ele distribui os valores dos graus de pertinência de forma normalizada através das várias classes baseado em um agrupamento natural das classes. Por outro lado, não é possível prever o tempo de processamento e alguns subconjuntos fuzzy podem ser não conectados. Outro problema é o fato de que devemos especificar *a priori* o número de classes, o que geralmente é uma tarefa complexa. Métodos de validação são em geral usados para aferir a qualidade das partições obtidas. Um outro problema em relação ao método é o custo computacional que é bastante elevado. Da mesma forma que o *k-means* convencional o FKM apresenta bons resultados apenas quando os agrupamentos são hiper-esféricos e possuem aproximadamente o mesmo número de objetos em cada classe. Quando não é este o caso, podem-se usar variantes do FCM, por exemplo o algoritmo G-K (Gustavson-Kessel) obtido pela distância escalonada de Mahalanobis no FKM. Maiores detalhes, e outros algoritmos relacionando métodos de CA com lógica nebulosa podem ser vistos em Bezdek (1981), Bezdek & Pal (1992), Krishnapuram e Keller (1994), Pal (1994), e Klir e Yuan (1995).

2.7 Modelos baseados em misturas de densidades de probabilidades

Modelos de misturas têm sido usadas em reconhecimento de padrões em diversas situações, como por exemplo, na estimação de densidades (Duda et al., 1998), no projeto de redes neurais do tipo RBF (*radial-basis-function*) (Haykin, 1999). Um modelo de mistura ou distribuição consiste de múltiplas funções densidade de probabilidade. Estas funções são denominadas densidades componentes da mistura, e são caracterizadas por diversos parâmetros desconhecidos. Referências dedicadas a este modelo incluem Everitt & Hand (1981), Titterington et al. (1985) e McLachlan & Basford (1988).

Apesar das densidades poderem ter formas paramétricas diferentes, geralmente é consideradas uma única forma pelo motivo da complexidade matemática e computacional. O problema do modelo de misturas envolve a estimação de vários parâmetros desconhecidos a partir de um conjunto de padrões. No caso de assumirmos a forma paramétrica de densidades normais, mais comum para este tipo de problema, precisamos estimar $(p+1)\cdot(p+2)\cdot K/2$ parâmetros, onde p é o número de variáveis, ou a dimensão dos padrões, $K \geq 1$ é o número de componentes. Tais parâmetros envolvem os vetores de média e as matrizes de covariâncias de cada componente, e suas respectivas probabilidades na mistura. Se assumirmos que cada padrão $\boldsymbol{x}_i$ tem probabilidade π_k de originar-se da sub-população k, $k \in (1, 2, \ldots, K)$, então descrever $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$, como uma amostra de

$$f(\boldsymbol{x}) = f(\boldsymbol{x}; \pi, \mu, \Sigma) = \sum_{k=1}^{K} \pi_k \cdot g_k(\boldsymbol{x}; \mu_k, \Sigma_k), \tag{2.44}$$

onde $\pi_k = (\pi_1, \pi_2, \ldots, \pi_K)$ são os K fatores de proporção da mistura, sujeitos a

$$0 \leq \pi_k \leq 1 \quad \text{e} \quad \sum_{k=1}^{K} \pi_k = 1, \tag{2.45}$$

e onde $g_k(\boldsymbol{x}; \mu_k, \Sigma_k)$ é a k-ésima função densidade componente, no nosso caso uma normal multivariada, dada por

$$g_k(x;\mu_k,\Sigma_k) = (2\pi)^{-p/2}|\Sigma_k|^{-1/2} exp\left[-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)\right]. \quad (2.46)$$

Vários autores usaram o modelo de misturas de densidades normais como modelo estatístico de problemas de agrupamentos, por exemplo Hartigan (1977), Binder (1978), Everitt (1993), etc. De (2.45) vemos que existem (K-1) fatores de proporção independentes. Estimar os parâmetros de $f(x)$, em (2.44), pode ser visto como a busca pelo processo estatístico que originou o conjunto de padrões. A figura 2.5 ilustra um processo Gaussiano no espaço $\Re^2$. Vemos que há uma maior concentração de pontos no maior eixo da elipse, ou maior variabilidade dos dados. Pode-se ver, também, na figura 2.5, elipses descrevendo pontos eqüidistantes do vetor de médias, usando a métrica de Mahalanobis. A figura 2.6 ilustra, em 3D, o modelo de geração dos dados da figura 2.5 e que deve ser recuperado através das técnicas de misturas de densidades.

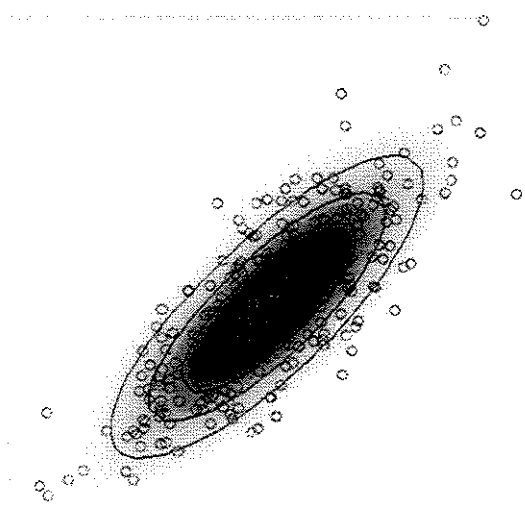


Figura 2.5: 300 pontos de um processo Gaussiano no espaço $\Re^2$.

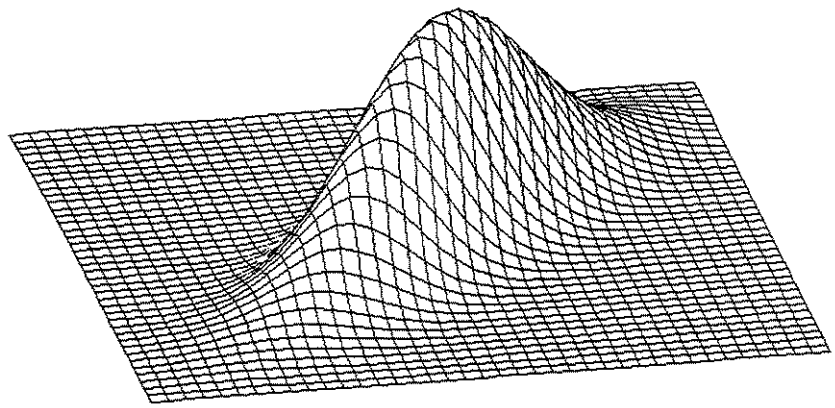


Figura 2.6: Modelo de geração dos dados da figura 2.4 em 3D.

Uma regra empírica (Hartigan, 1975) para adequação do modelo seria que o tamanho do conjunto de dados fosse maior que o número de parâmetros a serem estimados, i.e., $n > (p+1)\cdot(p+2)\cdot K/2$. Podemos classificar modelos de misturas em função da forma de suas matrizes de covariâncias. No caso mais geral as matrizes de covariâncias são diferentes para cada componente da mistura. O espaço de parâmetros para este modelo poderia ser escrito como

$$\Theta = \{\theta : \theta = (\pi_1, \mu_1, \Sigma_1, \pi_2, \mu_2, \Sigma_2, \ldots, \pi_k, \mu_k, \Sigma_k)\}. \qquad (2.47)$$

Casos mais simples, e menos flexíveis, podem ser modelados a partir de uma matriz de covariância que seja igual para os K componentes, i.e., $\Sigma_k = \Sigma$, $k = 1, 2, \ldots K$. Pode-se ainda simplificar a matriz de covariâncias, tornando-as diagonais e iguais para os vários componentes, i.e., $\Sigma_k = diag(\sigma_1^2, \ldots, \sigma_p^2)$, $k = 1, 2, \ldots K$. O modelo mais simples assume que todas as variáveis possuem a mesma variância, impondo no espaço p-dimensional uma geometria hiper-esférica, i.e., $\Sigma_k = \sigma^2 \cdot I$, $k = 1, 2, \ldots K$, onde I é a matriz identidade. A escolha por um modelo complexo, ou mais simples, ligado diretamente à escolha da forma das matrizes de covariância, deve ser feita de modo criterioso, levando-se em conta que conjuntos de dados relativamente pequenos podem levar o modelo a uma sobre-parametrização, por exemplo quando usando o modelo mais flexível (eq. 2.46).

A estimação dos parâmetros das equações (2.44-2.46) pode ser feita usando-se o algoritmo *expectation-maximization* (EM), desenvolvido por Dempster et al. (1977). Uma boa revisão do método é apresentada em Redner & Walker (1984), ver também em Bishop (1995). O EM maximiza a verossimilhança das amostras para um dado modelo de misturas. Para

escolher o número de componentes da mistura usam-se critérios de teoria de informação (ver seção 4.1.3).

O logaritmo da verossimilhança dos dados $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$, pode ser escrito como

$$l(\theta) \equiv \log L(\theta \mid X) = \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{K} \pi_k \cdot (2\pi)^{-p/2} |\Sigma_k|^{-1/2} exp\left[-\frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) \right] \right\} \quad (2.48)$$

A obtenção dos estimadores de máxima verossimilhança para todos os parâmetros desconhecidos é feita usando-se cálculo diferencial de matrizes, computando-se as derivadas parciais da função logaritmo da verossimilhança, $l(\theta)$, em relação a π_k, os vetores das médias μ_k, e Σ_k^{-1}. Fazendo as derivadas parciais em relação a estes parâmetros iguais a zero, temos, após um pouco de álgebra,

$$\hat{P}(k \mid \boldsymbol{x}_i) = \frac{\hat{\pi}_k \cdot g_k(\boldsymbol{x}; \hat{\mu}_k, \hat{\Sigma}_k)}{\sum_{k=1}^{K} \hat{\pi}_k \cdot g_k(\boldsymbol{x}; \hat{\mu}_k, \hat{\Sigma}_k)}, \qquad k = 1, 2, \ldots K. \quad (2.49)$$

$$\hat{\pi}_k = \frac{1}{n} \sum_{i=1}^{n} \hat{P}(k \mid \boldsymbol{x}_i), \qquad k = 1, 2, \ldots K. \quad (2.50)$$

$$\hat{\mu}_k = \frac{1}{n\hat{\pi}_k} \sum_{i=1}^{n} \boldsymbol{x}_i \cdot \hat{P}(k \mid \boldsymbol{x}_i), \qquad k = 1, 2, \ldots K. \quad (2.51)$$

$$\hat{\Sigma}_k = \frac{1}{n\hat{\pi}_k} \sum_{i=1}^{n} \hat{P}(k \mid \boldsymbol{x}_i) \cdot (\boldsymbol{x}_i - \hat{\mu}_k) \cdot (\boldsymbol{x}_i - \hat{\mu}_k)^T, \qquad k = 1, 2, \ldots K. \quad (2.52)$$

onde $\hat{\pi}_k$ é um estimador do fator de proporção da mistura π_k, $\hat{\mu}_k$ é o estimador do vetor de médias μ_k, e $\hat{\Sigma}_k$ é o estimador da matriz de covariâncias Σ_k. $\hat{P}(k \mid \boldsymbol{x}_i)$ é a probabilidade *a posteriori* estimada da pertinência do padrão $\boldsymbol{x}_i$ ao agrupamento k.

Alguns métodos foram propostos na literatura para resolver as equações 2.49-2.52, por exemplo usando o método de Newton-Raphson ou o algoritmo EM, com pequenas modificações.

Um algoritmo iterativo para o EM é descrito a seguir, adaptado de Bozdogan (1993):

Passo 1: Inicialização de parâmetros:

$$\pi_k^{(0)} = 1/K,\ \mu_k^{(0)} = \hat{\mu}_k\ ,\ \text{e}\ \Sigma_k^{(0)} = \hat{\Sigma}_k\ ,\ \text{para}\ k = 1, 2, \ldots K.$$

Para inicializar os vetores de média e as matrizes de covariâncias, neste trabalho, usou-se resultados do método *k-means* (Hartigan, 1975).

Passo 2: Estimação: Para $k = 1, \ldots, K$; $i = 1, \ldots, n$

Calcule as probabilidades a posteriori, $\hat{P}(k \mid \boldsymbol{x}_i)$, de acordo com a equação (2.49).

Passo 3: Maximização: Usando as equações (2.50 - 2.52), calcula-se $\hat{\pi}_k$, $\hat{\mu}_k$ e $\hat{\Sigma}_k$.

Passo 4. Os ciclos de iteração (passos 2 e 3) devem prosseguir até a convergência dos valores de π_k, μ_k, e Σ_k, ou até o atendimento de um outro critério de parada, como descrito abaixo.

Critério de parada: Caso a função logaritmo da verossimilhança cresça menos que um valor previamente especificado, ε, por exemplo 0.01, o processo pode ser interrompido.

Classificação: Após a determinação das estimativas de máxima verossimilhança, pode-se considerar cada distribuição como um agrupamento diferente, e os padrões podem ser atribuídos aos agrupamentos usando-se a regra de alocação de Bayes. O padrão $\boldsymbol{x}_i$ irá pertencer ao k-ésimo componente da mistura caso

$$\hat{\pi}_k \cdot g_k(\boldsymbol{x}; \hat{\mu}_k, \hat{\Sigma}_k) \geq \hat{\pi}_l \cdot g_l(\boldsymbol{x}; \hat{\mu}_l, \hat{\Sigma}_l), \quad \text{para todo}\ l \neq k. \tag{2.53}$$

Isto é equivalente a classificar o padrão $\boldsymbol{x}_i$ na mistura k para o qual a probabilidade estimada a posteriori de pertinência ao agrupamento, $\hat{P}(k \mid \boldsymbol{x}_i)$, seja a máxima.

2.8 Representação distribuída de protótipos dos agrupamentos

Métodos como o *k-means* usam protótipos com influência espacial isotrópica, o que leva a geometria dos agrupamentos serem hiper-esferas no espaço p-dimensional. No caso de misturas de densidades normais, a representação também é concentrada, porém a influência espacial pode não ser isotrópica, dependendo da matriz de covariâncias de cada protótipo. De qualquer forma, a geometria do agrupamento também é fixa, i.e., uma hiper-elipsóide, sendo a flexibilidade restrita ao ajuste dos parâmetros (π_k, μ_k, e Σ_k., $k = 1, 2, \ldots, K$). Além do formato geométrico dos agrupamentos (relativamente fixo) imposto no espaço pelos métodos particionais e pelos métodos baseados em misturas, a escolha errada do número de agrupamentos leva a grandes erros de classificação, por subdividir, em geral, um dado agrupamento em um ou mais agrupamentos, quando $K_{escolhido} > K_{ideal}$, ou fusão de dois ou mais agrupamentos em um, $K_{escolhido} < K_{ideal}$.

As figuras 2.7-a e 2.7-b ilustram a área (ou volume) de influência de dois protótipos nos espaços $\Re^2$ e $\Re^3$. Caso tivéssemos apenas um protótipo, $\boldsymbol{r}_i$, isolado, sob a métrica Euclidiana, sua influência no espaço $\Re^3$ vai até o infinito, ou seja, todos os pontos fariam parte do agrupamento sendo representado por $\boldsymbol{r}_i$ (figura 2.8). A partir do momento onde tenhamos dois protótipos, $\boldsymbol{r}_i$ e $\boldsymbol{r}_j$, podemos pensar em termos da quantização do espaço p-dimensional. Qualquer ponto $\boldsymbol{x}$ mais próximo do protótipo $\boldsymbol{r}_i$ será considerado como integrante da região dos pontos neste espaço p-dimensional sob a influência de $\boldsymbol{r}_i$. Pode-se então definir a área de influência do protótipo $\boldsymbol{r}_i$, no espaço $\Re^2$ (volume ou hipervolume de influência, para os casos no espaço $\Re^3$ ou espaço $\Re^p$, onde $p > 3$, respectivamente), como $\mathbf{A}(\boldsymbol{r}_i)$ tal que

$$\forall \boldsymbol{x} \in \mathbf{A}(\boldsymbol{r}_i) \Leftrightarrow \left\| \boldsymbol{x} - \boldsymbol{r}_i \right\|^2 < \left\| \boldsymbol{x} - \boldsymbol{r}_j \right\|^2, \quad i \neq j, \quad j = 1, 2, \ldots K. \tag{2.54}$$

onde K é o número de agrupamentos. Pontos que estejam equidistantes de dois ou mais protótipos estão na fronteira dos agrupamentos. Fronteiras serão formadas por hiperplanos no espaço $\Re^p$, que passam ortogonalmente ao segmento de reta que une os dois protótipos ($\boldsymbol{r}_i$ e $\boldsymbol{r}_j$). No caso em que consideremos a mesma importância de todos os protótipos, o hiperplano passa no meio deste segmento de reta. A quantização do espaço $\Re^p$ como

descrita é então uma generalização do diagrama de Voronoï. Na figura 2.7-a os protótipos r_i e r_j são, respectivamente, (0, 0, 0) e (1, 1, 0). Nesta figura, o volume de influência de cada protótipo, por razões de claridade, foi limitado em raio *1*. Na prática, a equação (2.54) estabelece que o volume ou área de cada protótipo vai até o infinito, ou até que esta influência toque no volume do protótipo vizinho, onde existirá uma fronteira de pontos não atribuídos a nenhum dos agrupamentos.

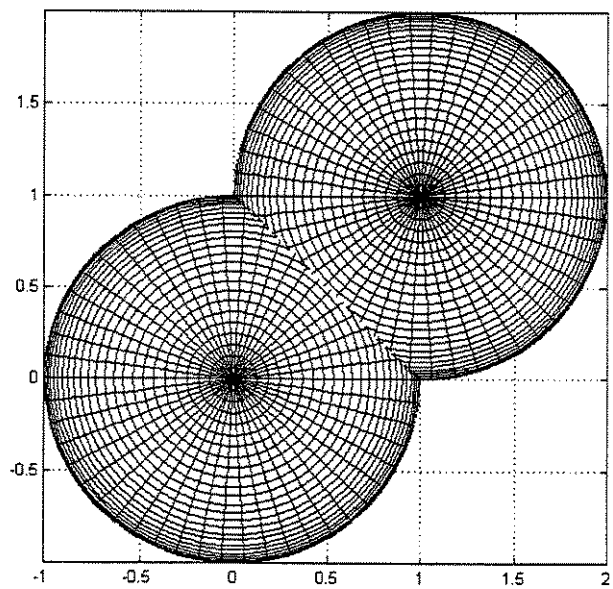


Figura 2.7-a: Influências espaciais de dois protótipos no $\Re^2$: r_i e r_j são, respectivamente, (0, 0) e (1, 1).

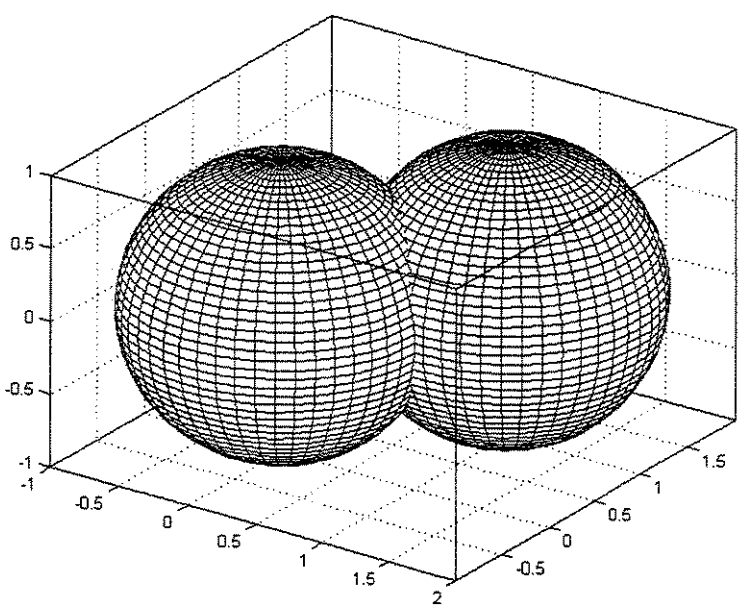


Figura 2.7-b: Influências espaciais de dois protótipos no $\Re^3$: r_i e r_j são, respectivamente, (0, 0, 0) e (1, 1, 0).

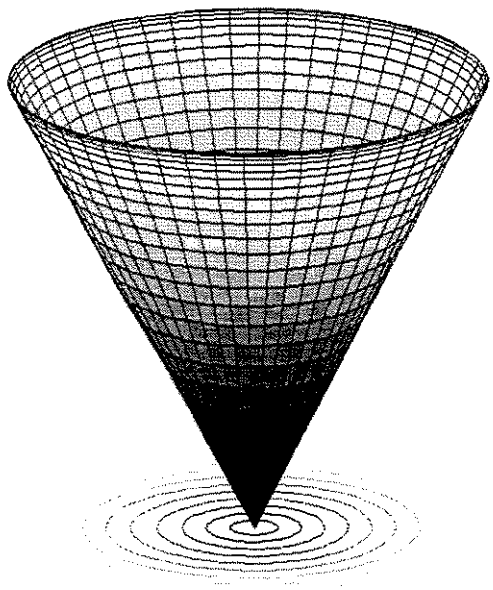


Figura 2.8-a - Distância Euclidiana de um protótipo no espaço $\Re^2$ plotada como altura (eixo z) e linhas de contorno ilustrando a influência equidistante no $\Re^2$.

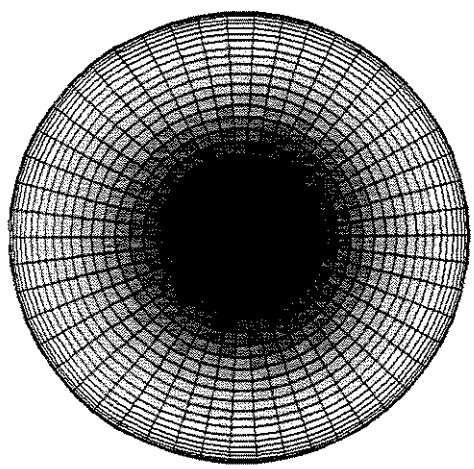


Figura 2.8-b - Influência de um protótipo no espaço $\Re^2$ representada por intensidades de cinza. Regiões mais claras representam pontos mais distantes do protótipo.

Nesta tese buscou-se métodos que descobrissem automaticamente o número de agrupamentos, em um dado conjunto de dados, e também que estes agrupamentos não possuíssem uma forma geométrica fixa *a priori* (ver capítulos 5-7). O objetivo é permitir que o sistema seja o mais flexível possível, descobrindo agrupamentos com geometrias quaisquer, e obtendo resultados mais próximos ao ideal. A idéia é usar um conjunto de

protótipos para representar um agrupamento, no lugar de termos um único representante, como é o caso do *k-means*. Assim, cada agrupamento C_k possui um conjunto de representantes $R_k = \{ r_1, r_2, \ldots, r_q \}$, onde $k = 1, 2, \ldots K$ e $q = 1, 2, \ldots | R_k |$, e onde $| \cdot |$ é a cardinalidade de um determinado conjunto de protótipos. Note que cada elemento do conjunto R_k é um vetor no espaço p-dimensional.

Assumiremos que cada sub-protótipo $\boldsymbol{r}_q$ será membro exclusivo de um agrupamento, i.e., se $\boldsymbol{r}_i \in R_i$, $\boldsymbol{r}_i \notin R_k$, $\forall\ k \neq i$, $k = 1, 2, \ldots K$. Associado ao conjunto de protótipos em cada agrupamento descrito por R_k pode-se ter ainda um conjunto de *links* entre os elementos $\boldsymbol{r}$ descrevendo um padrão de conectividade dentro do agrupamento.

A figura 2.9 ilustra um conjunto de dados não esférico e não Gaussiano. Ele foi gerado adicionando-se ruído uniforme a um semi-ciclo da função seno. Foram gerados 3000 pontos e o ruído possui média zero e valores mínimo e máximo -0.1 e 0.1, respectivamente.

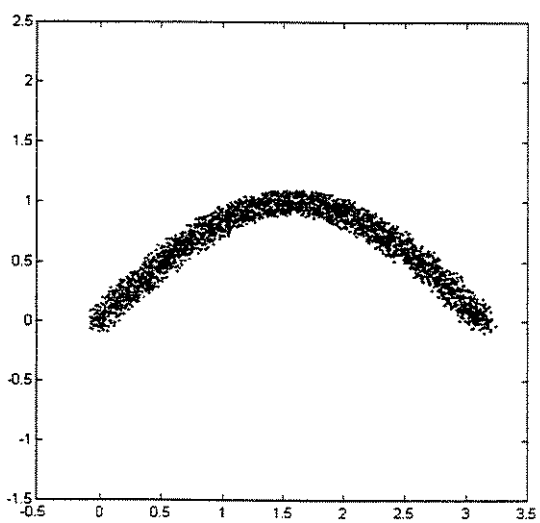


Figura 2.9: Conjunto de dados para testes, gerados artificialmente.

Caso usássemos um único protótipo para representar tal conjunto de dados, supondo que houvesse apenas um agrupamento, e considerando que este protótipo estivesse localizado no centro de massa dos dados, teríamos uma influência espacial como a apresentada na figura 2.10.

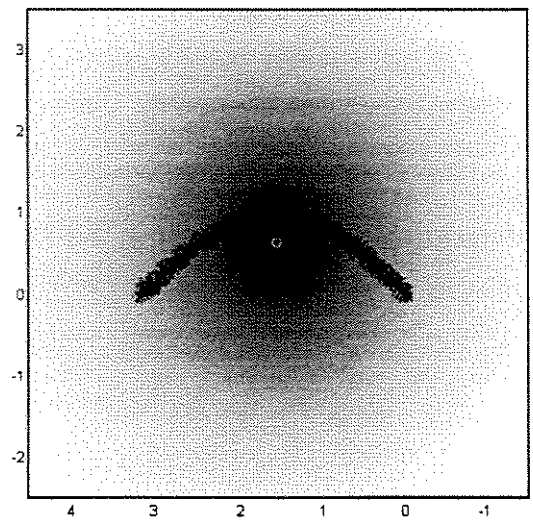


Figura 2.10 - Modelo concentrado - único protótipo no centro de massa dos dados

Uma forma de melhorar a influência espacial e representabilidade do agrupamento é usar um modelo distribuído, como descrito anteriormente, que levasse a uma modelagem mais próxima da densidade de probabilidade dos dados, e que pudesse ser mais flexível. Na figura 2.11 tem-se uma idéia de como seria um modelo distribuído para tal conjunto de dados. Mesmo que cada sub-protótipo tenha influência espacial isotrópica, a influência do agrupamento, que é a integral no espaço p-dimensional das influências de cada sub-protótipo, não é mais isotrópica. Vemos que ao modelar a estrutura de dados, ou o agrupamento, usando um modelo distribuído, obtemos uma enorme flexibilidade em relação ao modelo convencional, concentrado.

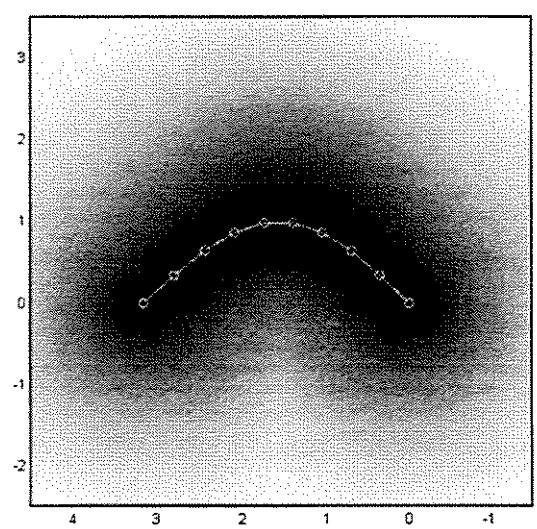


Figura 2.11: Influências espaciais do modelo distribuído e os sub-protótipos, em branco.

Um outro aspecto que pode ser considerado é que o erro de quantização é menor no caso apresentado na figura 2.11 do que para o caso da representação concentrada. Tal modelo será implementado no algoritmo SL-SOM (capítulo 5) onde um conjunto de neurônios pertencentes ao mesmo agrupamento serão rotulados conjuntamente.

A figura 2.12 ilustra o modelo obtido limitando o raio de influência de cada sub-protótipo do agrupamento em 0.5. Observa-se grande flexibilidade em relação ao modelo concentrado. A figura 2.13 apresenta a sobreposição dos dados originais ao agrupamento obtido via modelo distribuído.

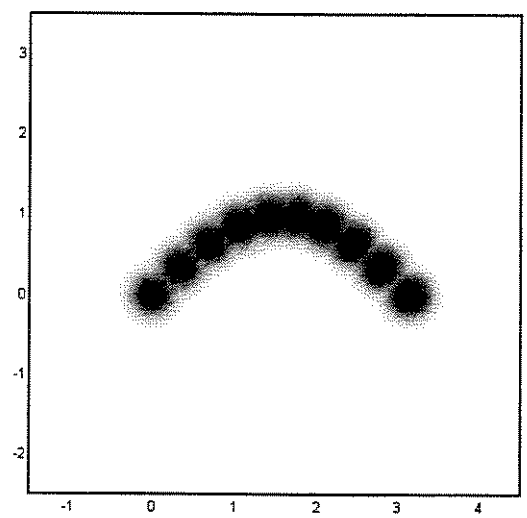


Figura 2.12 : Influências espaciais de cada sub-protótipo do modelo distribuído, limitadas a raio 0.5.

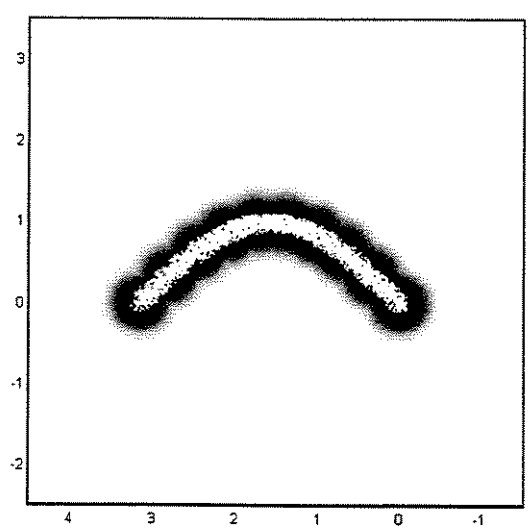


Figura 2.13 : Dados originais (em branco) sobrepostos à influência espacial conjunta do agrupamento no modelo distribuído, limitada a raio 0.5.

2.9 Sobre o número de agrupamentos

Talvez uma das escolhas mais difíceis em CA seja escolher o número adequado de agrupamentos em um conjunto de dados X. Idealmente, não existe um valor específico, K, que seja ótimo, ou regra geral para determiná-lo. Tal valor dependerá sempre do método que estamos empregando para descobrir os agrupamentos, além da escala que estamos interessados.

Freqüentemente usam-se técnicas de redução da dimensionalidade, como análise de componentes principais, para poder tentar chegar visualmente a um valor satisfatório. A escolha errada em alguns métodos, como o *k-means* e o *fuzzy k-means*, podem resultar em divisões ou fusões de agrupamentos, dependendo se o valor K foi escolhido acima ou abaixo do valor satisfatório. Geralmente pode-se aplicar a estes métodos um critério de validação para detectar se a configuração final obtida tem coerência com a métrica utilizada. O problema é que a métrica implica em uma estrutura geométrica no espaço p-dimensional, e o critério de validação irá, desta forma, acusar a coincidência da geometria dos dados à geometria dos agrupamentos encontrados.

A mesma deficiência do *k-means* ocorre com as redes neurais competitivas convencionais. Como deve-se especificar *a priori* o número de neurônios, que representam as classes, temos o mesmo problema da imposição de estrutura aos dados quando o valor é escolhido de forma errada. Esta é uma das principais motivações das redes neurais competitivas incrementais e das redes neurais competitivas hierárquicas, que em geral partem de um número pequeno de neurônios e vão acrescentando-os, podendo também eliminá-los, de acordo com uma dinâmica pré-estabelecida. Algumas destas técnicas serão abordadas no capítulo 4.

Uma hipótese bastante considerada em CA é que as amostras do conjunto de dados sejam provenientes de uma mistura de densidades de probabilidades com formato conhecido, em geral Gaussianas multivariadas (Wolfe 1970, 1978; Duda et al., 1998; Hamad et al., 1996; Firmin et al., 1997). O uso de técnicas de identificação de modelos de misturas (Titterington et al. 1985; McLachlan e Basford 1988; Bishop, 1995) podem usar índices ou critérios de informação, como por exemplo o critério de informação de *Akaike*, AIC, (Akaike, 1974; Bozdogan, 1993), ver por exemplo no capítulo 4, que são baseados na estimativa de máxima verossimilhança. Em geral, são efetuados vários testes para diferentes valores de K dentro de uma faixa de valores prováveis e escolhe-se o que resulte na maximização do critério usado no índice. Em geral, é difícil estabelecer testes de hipóteses para validação de resultados de CA. Uma revisão sobre as deficiências nestas escolhas é apresentado em SAS (1996).

Nos métodos hierárquicos, há necessidade de saber onde parar os dendrogramas. Milligan & Cooper (1985) compararam trinta métodos existentes na literatura, até então, para estimação do número de agrupamentos usando quatro métodos aglomerativos. Vários conjuntos de dados artificialmente gerados foram usados nas simulações. Os autores destacaram os três melhores critérios: uma pseudo-estatística F desenvolvida Calinski & Harabasz (1974), uma estatística referida como $J_e(2)/J_e(1)$ por Duda & Hart (1973), que pode ser transformada em uma pseudo-estatística t^2, e o critério de agrupamento cúbico (CCC). Os autores ainda sugerem análise de combinações de critérios, como por exemplo picos locais de CCC e da pseudo-estatística F combinados com valores baixos da pseudo-estatística t^2 e seguido de um valor elevado desta última na próxima fusão de agrupamentos. Os autores ainda enfatizam que os critérios são apropriados preferencialmente em agrupamentos compactos ou ligeiramente compridos, como é o caso dos provenientes de processos Gaussianos multivariados. Análises recentes de critérios são apresentadas em Gordon (1996).

O método empregado nesta tese (capítulo 5) será baseado na estabilidade de regiões conectadas de uma imagem de gradiente dos p-componentes de um mapa auto-organizado de Kohonen. Considerando cada neurônio como o representante de um micro-agrupamento, este método pode ser visto como uma sobre-estimação do número de agrupamentos que são posteriormente agrupados através de um processo de segmentação e rotulação de imagens. O resultado é que os agrupamentos possuirão um ou mais neurônios representantes, o qual irá permitir inclusive detecção de agrupamentos com geometrias arbitrárias, implementando, assim, o modelo de protótipos distribuídos, apresentado na seção anterior.

2.10 Técnicas relacionadas com classificação automática

Técnicas relacionadas com classificação automática incluem análise de componentes principais, análise discriminante e análise fatorial.

Análise de componentes principais (PCA) objetiva, principalmente, reduzir o conjunto original de variáveis para um conjunto menor através de combinações lineares destas com o intuito de formar novas variáveis que possuam variância máxima e que sejam não-correlatas, i.e., ortogonais. Pelo fato de que, em geral, muitas das variáveis em um conjunto de dados possam ser correlacionadas, ou que possuam pequena variabilidade, o uso de PCA pode permitir grande redução da informação dos dados originais em algumas poucas combinações. Busca-se reduzir o conjunto de variáveis a 2 ou 3 componentes principais

com o objetivo de poder visualizar a estrutura dos dados e facilitar a interpretação por parte do usuário. Pode-se, ainda, efetuar agrupamentos a partir das projeções dos dados nos componentes principais. Geometricamente, obtemos uma combinação linear que representa uma seleção de um novo sistema de coordenadas obtidas pela rotação do sistema original nos eixos coordenados. Análise fatorial está, de certa forma relacionada a técnicas de PCA. Os objetivos são semelhantes, porém em análise fatorial busca-se fatores que expliquem os dados. Além disso, a formulação matemática do problema é diferente de PCA. Boas introduções a PCA e análise fatorial podem ser vistas em Jobson (1991, 1992) e Johnson e Wichern (1998).

Análise discriminante (AD), por outro lado, objetiva a classificação de objetos em classes mutuamente exclusivas e exaustivas. A principal diferença de AD para CA é que na primeira uma regra de classificação é modelada a partir de um conjunto de dados de treinamento rotulado. Supõe-se a existência de classes *a priori*, e o objetivo é maximizar a generalização ou acertos em um conjunto de dados não usado no treinamento, ou conjunto teste. Podem ocorrer situações em que técnicas de AD e CA possam ser empregadas conjuntamente. Pode-se desejar verificar a existência de agrupamentos e / ou estabelecer regras de classificação dos objetos nas classes descobertas. O termo classificação tem sido usado tanto em análise de agrupamentos, por exemplo (Gordon, 1981), quanto em análise discriminante, como por exemplo (Duda et al., 1998).

Não temos o propósito de estender a abordagem desses assuntos, apesar de considerarmos de grande importância. Boas fontes de consultas incluem livros e textos sobre análise estatística multivariada como apresentado em Mardia et al. (1979), Jobson (1991, 1992) e Johnson e Wichern (1998). Métodos de análise discriminante usando redes neurais foram abordados pelo autor em Costa (1996a-c), Costa & Gonzaga (1996a-b) e em Costa & Netto (1997).

2.11 Sumário

Cada método de classificação, hierárquica ou não, possui uma geometria própria que toma forma de acordo com a escolha dos parâmetros e dos dados usados para estimá-los. O sucesso da aplicação de um determinado algoritmo, ou método, depende, na maioria das vezes, da coincidência (ou aderência) da geometria natural dos dados à geometria que o método impõe no espaço p-dimensional. Caso contrário, estaremos impondo uma estrutura aos dados, ao contrário de estar recuperando-a. O problema de classificação automática é motivante por muitas razões, dentre as quais:

Simplicidade aparente. Em princípio, a idéia é bastante simples, i.e., subdivida os dados em subgrupos que sejam mais similares entre si do que aos outros dados alocados para os outros grupos.

Complexidade inerente às várias escolhas como: atributos, critérios de similaridade, função objetivo, parâmetros dos algoritmos, dimensionalidade dos dados, etc. Tais escolhas podem, e geralmente atuam como um processo semi-supervisionado na busca de uma solução, que pode, ou não, levar a um resultado satisfatório.

***Quantidade de aplicações possíveis*:** pode-se aplicar os métodos de classificação a qualquer área do conhecimento humano, sejam bancos de dados de empresas, imagens de satélites ou médicas, em pesquisas biológicas, ciências sociais, antropologia, etc.

No caso específico de engenharia de sistemas inteligentes e de computação, o uso de técnicas de classificação automática pode favorecer, por exemplo, o surgimento de novos métodos de organização e recuperação de informações, de forma mais natural e com maior eficiência (menos tempo de acesso e menor espaço de memória ocupados).

Advogamos em favor dos modelos distribuídos em relação aos modelos concentrados de representação dos agrupamentos devido ao grande ganho de flexibilidade na geometria de tais agrupamentos. O custo disso é ter que dispor de mais protótipos (ou sub-protótipos) envolvidos em cada agrupamento e *links* conectando-os como um grafo no espaço *p*-dimensional. A grande vantagem, que ficará evidente nos capítulos posteriores, é que podemos fazer isto usando redes neurais auto-organizáveis (Costa & Netto, 1998, 1999a,c), fazendo análise de redes SOM treinadas. Exemplos de métodos discutidos neste capítulo serão mostrados nos próximos capítulos.

Capítulo 3

Mapas Auto-organizáveis de Kohonen

Este capítulo descreve o tipo básico de rede neural artificial usado neste trabalho. Os mapas (de atributos) auto-organizáveis (*Self-Organizing Maps* - SOM) foram inicialmente desenvolvidos pelo Prof. Teuvo Kohonen (1982a,b) e constituem um dos principais paradigmas na área de redes neurais artificiais. O SOM foi inspirado no modo pelo qual informações sensoriais são mapeadas no córtex cerebral. SOM é um algoritmo não supervisionado que aproxima a densidade de probabilidade dos estímulos de entrada ao mesmo tempo em que reduz a dimensionalidade, tentando preservar ao máximo as relações topológicas entre os dados.

O objetivo deste capítulo é fornecer fundamentos para estudo e aplicação do SOM nos problemas de classificação automática de dados. O SOM é uma ferramenta básica junto a qual os algoritmos desenvolvidos nesta tese são aplicados. Descrevemos a arquitetura e o processo de treinamento, analisamos o funcionamento e discutimos aspectos de convergência do algoritmo. Também discute-se uma série de limitações deste modelo e algumas heurísticas para tentar resolvê-las. Haveria necessidade de espaço de um ou mais livros para descrever o SOM e suas principais variantes. Neste capítulo, restringimo-nos a aspectos mais fundamentais. Uma vasta literatura está disponível, sendo Kohonen (1997a) a principal referência na atualidade.

3.1 Motivação Biológica

O córtex cerebral humano é uma fina camada de células, de aproximadamente um metro quadrado, contendo seis camadas de neurônios redobradas para caber no crânio (Freeman, 1991). Apesar da mecânica e dos processos do córtex não serem ainda completamente entendidos, evidências anatômicas e fisiológicas sugerem a existência de iteração lateral entre os neurônios. No caso do cérebro dos mamíferos, ao redor de um centro de excitação existe uma região (50 a 100 μm) de excitação lateral. Ao redor desta existe uma área de ativação inibitória, aproximadamente de 200 a 500 μm (Kohonen, 1984). Novamente, ao redor desta última área, segue-se uma região de excitação fraca (alguns centímetros). O fenômeno de iteração lateral pode ser modelado pela função *chapéu-mexicano (Mexican-*

hat), figura 3.1, que matematicamente pode ser descrita pela diferença entre duas Gaussianas.

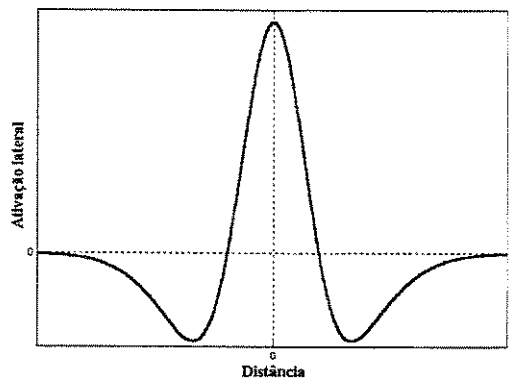


Figura 3.1: Função Chapéu-mexicano descrevendo a ativação lateral

Reconhecimento de padrões (RP) está ligado, entre outras coisas, à memória. Uma solução encontrada durante o processo evolutivo das espécies foi desenvolver representações internas eficientes de estímulos sensoriais. Atualmente sabe-se que o cérebro possui áreas especializadas para processar diferentes modalidades de sinais. Nestas áreas, principalmente nas áreas dedicadas a processamento primário de sinais sensoriais, os neurônios respondem às muitas qualidades dos estímulos de uma forma ordenada. Por exemplo, na área auditiva, existe uma 'escala' para freqüências acústicas diferentes. Na área visual, existem mapas para processar a orientação de segmentos de reta, mapas de cores (Zeki, 1993), etc. Mapas ordenados topograficamente, geralmente bidimensionais, e o conceito de formação de imagens abstratas das dimensões das características sensoriais aparentam ser um dos mais importantes princípios na formação das representações internas no cérebro (Kohonen, 1989b).

Talvez o trabalho experimental mais influente nesta área foram os estudos de Hubel e Wiesel (1962) usando micro-eletrodos no córtex visual *1* de gatos. Estímulos semelhantes, como segmentos de reta com orientação variando poucos graus, excitaram neurônios próximos. Registrando as respostas mais intensas com um eletrodo, que foi inserido paralelamente à superfície do córtex e gradualmente movido ao longo do tecido nervoso, obtém-se uma série de orientações que em geral variam suavemente ao longo do córtex. A figura 3.2 mostra dados experimentais de Hubel e Wiesel, no qual vemos também descontinuidades ocasionais.

Percebe-se na figura 3.2 uma espécie de mapeamento, onde orientações similares excitam regiões do tecido nervoso próximos. No caso do córtex auditivo, experimentos indicam uma escala logarítmica de freqüências. Neurônios em posições diferentes são excitados por

sons diferentes e as posições relativas refletem de uma certa forma o relacionamento entre o conjunto de sons.

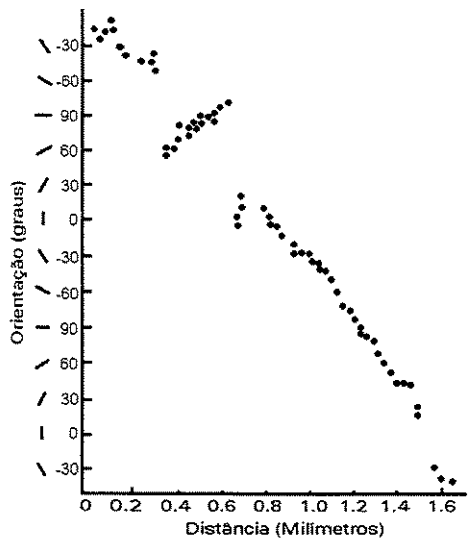


Figura 3.2 - Sensibilidade à orientação versus distância - Adaptado de Hubel e Wiesel (1962)

Apesar das idéias de auto-organização em sistemas neurais terem sido iniciadas no final dos anos 50 e início dos anos 60 (Yovits e Cameron, 1960), C. von der Malsburg demonstrou pela primeira vez, em 1973, a possibilidade de treinar uma rede neural usando métodos competitivos de forma a criar um mapeamento semelhante ao apresentado na figura 3.2. Modelos de auto-organização biologicamente inspirados foram desenvolvidos (Willshaw e von der Malsburg, 1976) baseados no estudo de neurônios que respondem seletivamente a estímulos, como os sensíveis a intensidade de luz e orientação de segmentos de retas, no córtex visual.

Kohonen (1982a,b) generalizou tal modelo no SOM. Previamente, Kohonen havia se dedicado a estudos sobre memória associativa e modelos para atividade neuro-biológica. A popularização do SOM deve-se a muitos fatores, como por exemplo o seu esforço de manter uma fonte de referências sempre atualizada. A simplicidade e ao mesmo tempo o poder computacional do SOM, aliados à disponibilidade do código fonte via *internet* (Kohonen et al., 1996a,b) são algumas das outras razões.

Como este trabalho não tem objetivo de aprofundamento nos aspectos do SOM relacionados com as estruturas neurofisiológicas, concluímos esta seção com uma lista de referências sobre o assunto, de forma que o leitor interessado possa se estender na matéria. Algumas referências incluem Bauer (1994), Boehme *et al.* (1994), Dedieu & Mazer

(1992), van Gils (1993), Kohonen (1994), Morasso & Sanguineti (1994); Obermayer *et al.* (1991, 1992), Pomierski *et al.* (1993), Saxon (1991) e Sutton (1994).

3.2 Redes neurais artificiais

Após a segunda metade dos anos 80 o interesse em redes neurais artificiais (RNAs) para reconhecimento de padrões foi restabelecido, após o trabalho de Rumelhart et al. (1986) com o algoritmo "*back propagation*" (BP) para treinamento de redes de múltiplas camadas. Atualmente, existem centenas de arquiteturas e algoritmos de treinamento de RNAs. Diversas características presentes nestas estruturas de processamento fazem delas umas das mais promissoras na área de reconhecimento de padrões (Bishop, 1995), (Ripley, 1996), (Haykin, 1999).

A idéia básica é que unidades de processamento simples computando certas funções matemáticas, dispostas em uma ou mais camadas, interagindo umas com as outras e com o ambiente, apresentam elevado poder computacional. Uma das vantagens em relação aos métodos estatísticos é que não necessitamos possuir densidades de probabilidade para cada classe de objetos. Embora não se saiba com profundidade, do ponto de vista matemático, o comportamento esperado para a maioria das classes de RNAs, tem-se constatado experimentalmente que muitas RNAs projetadas para servir como classificadores fornecem respostas na camada de saída que estimam a probabilidade *a posteriori* Bayesiana (Richard & Lippman, 1991). Para uma revisão descrevendo métodos de classificação envolvendo RNAs e classificadores estatísticos, ver Ripley (1994). Um modelo de rede fortemente baseado em estatística é a PNN (probabilistic neural networks) (Masters, 1995). Atualmente, a literatura mostra que RNAs podem ser modeladas de maneira que o resultado seja assintoticamente convergente aos métodos Bayesianos (Bishop, 1995).

RNAs são apropriadas para tarefas de percepção, como o reconhecimento, classificação e auto-associação de padrões. Apesar da maioria dos modelos atuais serem baseados mais em métodos numéricos que biológicos, a inspiração para diversas RNAs provém de áreas como a neurofisiologia. Atualmente a pesquisa em RNAs envolve diversas áreas do conhecimento, envolvendo pessoas com propósitos e formações diferentes. Para o projeto de classificadores, RNAs apresentam uma série de características desejáveis, tais como a tolerância ao ruído, a capacidade de generalização, o aprendizado adaptativo a partir de exemplos e processamento paralelo. Em ambientes industriais, diversas fontes de ruído, como interferências elétricas, variação de propriedades físicas nos componentes dos equipamentos de captação e processamento das imagens, erros de quantização ou binarização, etc., degradam os sinais, implicando em uma diminuição na taxa de acertos dos

sistemas de classificação e reconhecimento de padrões. Muitos desses fenômenos não são simples de equacionarmos, e dessa forma, a utilização de RNAs contribui para a robustez do sistema, pois todo treinamento é feito com exemplos de forma que a estrutura da rede se molda às características do problema.

RNAs são especificadas pela arquitetura da rede, características dos neurônios, dinâmica de processamento, e regra de treinamento ou aprendizado (RT). A RT especifica como os pesos devem ser adaptados durante o aprendizado para melhorar seu desempenho e a maioria delas possui um relacionamento próximo ao tipo de topologia da rede. Os métodos de aprendizado incluem pré-programação, regras de treinamento supervisionadas ou não supervisionadas. Geralmente, a alteração dos parâmetros de uma rede só é permitida durante a fase de treinamento, permanecendo estáticos durante as fases de teste e execução.

A maioria das redes tem sua arquitetura ligada diretamente à função, i.e., poucas são as redes de uso geral, que servem por exemplo para problemas de classificação, otimização, predição, etc.. Seguindo Maren et al. (1990), pode-se descrever as estruturas em diversos níveis: micro-estrutura, a menor estrutura de uma rede, i.e., o neurônio; meso-estrutura, a rede, no qual o projeto está relacionado com sua função; e a macro-estrutura, duas ou mais meso-estruturas trabalhando em conjunto para efetuar tarefas mais complexas.

Os neurônios tipicamente funcionam como integradores não lineares de sinais. Um neurônio coleta sinais de outros neurônios, integra os dados, subtrai de um valor chamado de limiar e passa o resultado através de uma função de ativação (ver figura 3.3). O resultado da operação é chamado de grau de ativação do neurônio, sendo este valor ponderado e coletado pelos neurônios da próxima camada. As características que diferenciam os neurônios em uma RNA são: (*i*) o tipo da função de ativação, que pode ser, por exemplo, sigmoidal, linear, de base radial (Gaussiana) e a função degrau; (*ii*) a natureza dos sinais utilizados para comunicação entre neurônios, que podem ser contínuos ou binários; (*iii*) a dinâmica dos neurônios: determinística ou estocástica; (*iv*) a adição de outros parâmetros ao neurônio, como limiares adaptativos, ganhos, a possibilidade de condução do potencial de ação por mais de um ciclo, etc. A equação 3.1 descreve o processamento efetuado por um neurônio simples como o mostrado na figura 3.3.

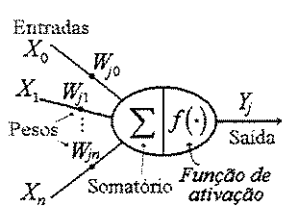


Figura 3.3: o neurônio artificial

$$Y_j = f\left(\sum_{i=1}^{n} w_{ij}.X_i - \theta_j\right) \tag{3.1}$$

onde X_i é a i-ésima entrada[1], w_{ij} é o peso da conexão entre o neurônio i de uma camada k ao neurônio j da camada $k+1$, θ_j é limiar do neurônio j da camada $k+1$, e Y_j é a resposta do neurônio j.

Matematicamente, o resultado do produto interno entre o vetor de entrada ($\boldsymbol{x}$) e o vetor de pesos do neurônio ($\boldsymbol{w}$) é mapeado através da função de ativação, produzindo a resposta do neurônio, Y_j. A função sigmoidal, $f(x)=\left(1+e^{-\lambda x}\right)^{-1}$, onde λ é um parâmetro que controla a inclinação da curva, é uma das mais utilizadas pois apresenta uma motivação biológica ao representar o efeito de saturação da resposta do neurônio. Além disso, o cálculo de sua derivada de primeira ordem é fácil de se determinar, o que implica em uma simplificação do esforço computacional do método de minimização do erro da rede, como ocorre no algoritmo *back-propagation*. Várias outras funções de ativação estão disponíveis na literatura, assim como algoritmos de otimização determinísticos, estocásticos ou evolutivos para estimação dos pesos, dada uma função de erro, ou função objetivo, para a rede (Bishop, 1995), (Duda *et al.*, 1998), (Haykin, 1999).

Meso-estruturas de RNAs estão relacionadas com a organização física - arranjo dos neurônios (Maren *et al.*, 1990). As características das meso-estruturas permitem a discriminação das redes em classes. As redes são compostas de diversos neurônios e podem possuir uma, duas ou várias camadas (multi-camadas). Os neurônios podem ser organizados em diversos padrões de conexão. Algumas redes permitem apenas o fluxo de sinais para frente (*feed-forward*) enquanto outras podem possuir conexões retroativas (*feedback*), laterais, conexões que pulam camadas, etc. A estrutura de uma RNA está, geralmente, relacionada com sua função. A saída da rede requer a ativação de um ou mais neurônios de saída. Estes resultados podem ser interpretados como uma classificação de padrões (heteroassociação) ou como uma versão completa do padrão de entrada (auto-associação), livre de ruído e distorções. Outro parâmetro da operação da RNA é o modo de operação: síncrono ou assíncrono. Para RNAs que necessitam funcionar em tempo real ou que possuam tamanho relativamente grande, este parâmetro se torna mais representativo. A figura 3.4 ilustra a arquitetura básica de uma rede de múltiplas camadas, bastante usada em reconhecimento de padrões e diversas outras aplicações, geralmente treinada com métodos

de aprendizado supervisionados. Aspectos deste tipo de rede e sobre seu treinamento podem ser vistos em Bishop (1995), Costa (1996a), e Haykin (1999).

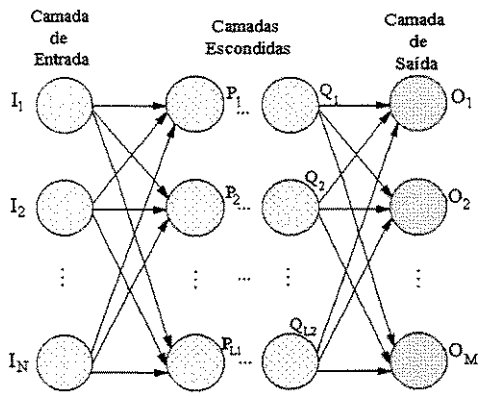


Figura 3.4: Arquitetura básica de uma rede de múltiplas camadas.

Muitos esforços têm sido empregados na tentativa de otimizar o treinamento de redes multicamadas. Costa (1996a) utilizou um sistema de treinamento híbrido, combinando métodos de minimização local, como os algoritmos gradiente conjugado e de Levenberg-Marquardt, com métodos de minimização global, como o *simulated annealing* (Costa, 1996b), (Costa e Gonzaga, 1996a,b).

Macro-estruturas são tópicos de pesquisa atual. Poderíamos utilizar uma rede especialista para extração de atributos e outra específica para classificação. Dividir tarefas complexas em específicas, utilizando módulos especializados de processamento parece ser uma estratégia utilizada pelos sistemas neurais biológicos (Zeki, 1993). O maior problema por enquanto é a integração das meso-estruturas que são geralmente treinadas separadamente e *off-line* para adquirirem habilidades específicas.

Kohonen (1990, 1997a) classificou em três categorias as RNAs de acordo com as arquiteturas e o tipo de processamento de sinais empregado:

1. *Signal-Transfer Networks*, que transformam um conjunto de sinais de entrada X em um conjunto de sinais de saída Y, $f : X \rightarrow Y$, onde a transformação f é paramétrica e definida por funções de base ajustadas por treinamento supervisionado. Exemplos incluem redes neurais de múltiplas camadas (Rumelhart et al., 1986), e redes de base radial (RBF) (Bishop, 1995).

[1] Ou o estado do neurônio i da camada precedente, caso hajam várias camadas e a camada atual não seja a primeira.

2. *State-Transfer Networks*, são redes recorrentes, onde o padrão de entrada define um estado inicial de atividade e após alguns estados de transição obtêm-se um estado estável, um atrator, que é identificado como o resultado da operação. Nesta categoria, estão por exemplo, as redes de Hopfield (Hopfield & Tank, 1986).
3. *Redes neurais competitivas*, onde todos os neurônios recebem, em geral, entradas iguais e neurônios vizinhos competem via interações laterais. Cada neurônio (ou grupo destes) desenvolve-se de forma adaptativa em detectores de padrões diferentes, atuando como um decodificador de diferentes domínios do espaço vetorial de entrada. Em geral são treinadas sem supervisão e são chamadas de redes auto-organizáveis. Nesta categoria incluem-se o modelo ART, *Adaptive Resonance Theory*, (Carpenter & Grossberg, 1987), e o SOM (Kohonen, 1982a; 1997).

Nesta tese nos concentramos nesta última categoria, as redes auto-organizáveis, mais especificamente no SOM. Outras arquiteturas, e o próprio desenvolvimento da área de redes neurais podem ser vistos em vários livros, como por exemplo, Haykin (1999). A seguir, descrevemos brevemente a idéia do aprendizado competitivo, e mais adiante, sua implementação no SOM.

3.3 Aprendizado Competitivo

O objetivo básico do aprendizado competitivo é fazer com que neurônios se especializem em estímulos apresentados de forma não supervisionada. Isto é, nenhuma informação sobre a classe do estímulo apresentado é usada no processo de ajuste dos pesos sinápticos. Todos os neurônios recebem o mesmo conjunto de entradas e competem, através de uma dinâmica que usa conexões laterais, com todos os outros neurônios. Estas conexões laterais podem ser positivas (no caso da auto-realimentação) ou inibitórias (negativas). Existem evidências da importância do aprendizado competitivo na formação de mapas topográficos no cérebro (Durbin *et al.*, 1989), (Ambros-Ingerson *et al.*, 1990).

O princípio básico de aprendizado competitivo é quantização vetorial (QV), que de uma forma ou de outra, aproxima a função densidade de probabilidade do sinal de entrada por um conjunto finito de vetores de referência, ou *codebooks*. Várias aplicações de QV estão disponíveis nas áreas de telecomunicações e processamento de sinais e imagens.

Existem três elementos básicos em regras de aprendizado competitivo (Rumelhart & Zipser, 1985):

1. O conjunto de neurônios com mesma função de ativação, apenas diferindo inicialmente pela aleatoriedade da distribuição dos pesos sinápticos, o que os farão responder diferentemente para um dado conjunto de padrões.
2. Os vetores de pesos que conectam neurônios entre as camadas de entrada e de saída são limitados, por exemplo, $\| \boldsymbol{m} \| = 1$.
3. Um mecanismo que permita aos neurônios competir pelo direito de responder a um dado subconjunto de entradas. No final do processo de competição, apenas um neurônio estará ativo. Esta é a regra *winner-takes-all.*

Tais condições são em muitos casos estendidas ou mesmo desconsideradas. Por exemplo existem regras onde podemos considerar mais de um neurônio vencedor para um dado padrão, ou mesmo não necessitar normalizar os pesos como descrito pela condição 2.

Considere a rede apresentada na figura 3.5. Suponha que um vetor $\boldsymbol{x}$, normalizado ($\|\boldsymbol{x}\| = 1$), é apresentado à camada de entrada da rede. Cada neurônio O_i, $i = 1, 2, \ldots k$, da camada de saída irá computar um valor de ativação. O neurônio vencedor, c, é o neurônio que apresenta a ativação máxima, i.e., maior similaridade com o padrão apresentado. As conexões laterais podem ser utilizadas em um processo dinâmico de competição (Kohonen, 1993), de forma a inibir os outros neurônios. No final do processo apenas um neurônio, c, permanecerá ativo. Existem várias formas de fazer isto, sendo a mais simples pegar o neurônio vencedor e saturá-lo, por exemplo, em 1, enquanto todos os outros são desativados, i.e., suas ativações são anuladas para aquele padrão.

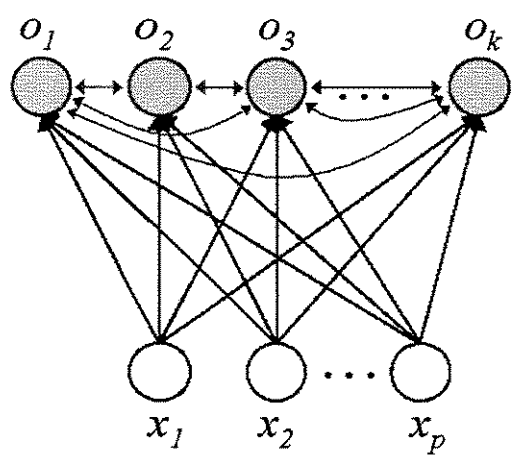


Figura 3.5: Esquema simplificado de uma rede neural competitiva

Apesar de algumas semelhanças com o método *k-means*, no caso do *k-means* a cada iteração os vetores de média são calculados de acordo com os membros provisórios de cada

agrupamento. No caso do aprendizado competitivo, o ajuste é restrito ao neurônio vencedor a cada apresentação de um padrão, sendo este neurônio vencedor o neurônio que possui o conjunto de pesos mais semelhante ao padrão apresentado. Dado um critério de distância, ou dissimilaridade, entre $\boldsymbol{x}$ e $\boldsymbol{m}_i$, $d(\boldsymbol{x}, \boldsymbol{m}_i)$, o vencedor, c, é identificado de forma que

$$d(\boldsymbol{x}, \boldsymbol{m}_c) = \min_i \{d(\boldsymbol{x}, \boldsymbol{m}_i)\}. \tag{3.2}$$

A adaptação ocorre apenas aos pesos sinápticos do neurônio vencedor, c:

$$\Delta \boldsymbol{m}_c(t+1) = \alpha(t) \cdot [\boldsymbol{x}(t) - \boldsymbol{m}_c(t)]. \tag{3.3}$$

Para todos os outros neurônios, $i = 1, \ldots, k$, $i \neq c$, $\Delta \boldsymbol{m}_i(t+1) = 0$. A taxa de aprendizado, $\alpha(t)$ pode ser uma função (ou seqüência de valores) decrescente, monotônica, limitada em $0 < \alpha(t) < 1$. O efeito desta regra é o deslocamento dos pesos do neurônio vencedor na direção do padrão $\boldsymbol{x}$ (veja a figura 3.6).

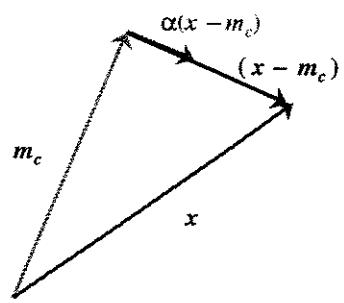


Figura 3.6: Esquema de atualização dos pesos em uma rede neural competitiva

3.4 Estrutura básica do SOM

Apesar de ter sido desenvolvido inicialmente para tentar modelar áreas sensoriais do córtex, e das iterações laterais entre os neurônios em tais estruturas, o SOM é um algoritmo extremamente simplificado e apenas superficialmente podemos compará-lo a uma estrutura biológica real. Porém, o nosso objetivo neste trabalho é a aplicação do mapeamento não linear que o algoritmo faz junto aos dados de entrada, de forma não supervisionada, e não o estudo de características reais de áreas do córtex.

A visão moderna do SOM é de uma ferramenta de software para a visualização de dados. Geralmente dados de aplicações reais são multi-dimensionais o que torna difícil a visualização das relações entre os dados. O SOM implementa uma projeção não-linear de um espaço de dados sensoriais ou de atributos $\Re^p$, geralmente de dimensionalidade elevada ($p >> 2$), em um conjunto discreto de neurônios, geralmente dispostos na configuração de um vetor ou uma matriz. Relações estatísticas complexas e não-lineares entre os dados são convertidas em relações geométricas simples sobre um *'display'* de menor dimensionalidade. Apesar de ter sido relacionado a componentes principais (CP), existem diferenças fundamentais entre estes dois métodos, como por exemplo o espaço discreto de saída do SOM, enquanto que em CP o espaço é contínuo. Além disto, componentes principais são obtidos por combinações lineares das variáveis, enquanto que o SOM é obtido matematicamente por regressão recursiva e não paramétrica (Kohonen, 1997a), e ao levar em conta a topologia da rede resulta em uma projeção não linear da densidade de probabilidade $p(x)$ dos sinais de entrada sobre a grade de neurônios.

Apesar de alguns autores descreverem como antagônicos, os dois objetivos principais do SOM são reduzir a dimensionalidade dos dados ao mesmo tempo em que se tenta preservar as relações métricas e topológicas do espaço de entrada. A dimensionalidade do mapa é dada pelas relações de vizinhança entre os neurônios. Uma outra característica obtida de um SOM treinado é a aproximação da função densidade de probabilidade dos dados, o que também será explorado neste trabalho, de uma forma ordenada. Devido à preservação das relações topológicas, informações semelhantes são mapeadas em neurônios próximos, eventualmente no mesmo neurônio, o que caracterizará a quantização ou agrupamento do espaço de entrada. As relações de vizinhança entre os neurônios nos permitirão a união posterior dos neurônios que respondem a estímulos semelhantes em conjuntos de neurônios. Esta é uma diferença básica frente a outros modelos de redes neurais competitivas, como a ART (Carpenter & Grossberg, 1987), no qual pode ocorrer de pontos vizinhos no espaço de entrada serem mapeados em neurônios distantes na camada de saída da rede.

Um mapa auto-organizável consiste de duas camadas de neurônios: a camada de entrada, I, e a camada de saída (ou de Kohonen), U. As entradas da rede são vetores no espaço p-dimensional, geralmente no espaço $\Re^p$. Cada neurônio i da camada de Kohonen possui um vetor também no espaço $\Re^p$ associado, $\boldsymbol{m}_i = [m_{i1}, m_{i2}, ..., m_{ip}]^T$. Os neurônios na camada de Kohonen são conectados aos neurônios adjacentes por uma relação de vizinhança que descreve a estrutura do mapa, como descrito anteriormente. No caso bidimensional podemos ter vizinhança tipo 4-conectados (mapas retangulares) ou 6-conectados (mapas hexagonais). A escolha da topologia, assim como o tamanho do mapa, $N \times M$, dependem da

aplicação. Geralmente usam-se SOMs retangulares por razões de simplicidade. A figura 3.7 ilustra um SOM para o caso em que $p = 3$, e a camada U possui tamanho *7x10*.

Dois tipos de distâncias existem no SOM: a primeira é a distância entre os índices da matriz ou vetor que forma o reticulado (*grid*) dos neurônios, *(k, c)*, onde k e c representam as coordenadas dos neurônios no *grid*. Esta distância é bastante usada durante o treinamento, no qual a atualização dos pesos sinápticos será efetuada, em um dado passo t, para neurônios dentro de um raio (decrescente com t) do neurônio vencedor (ver figura 3.8). A outra distância refere-se à distância no espaço de pesos, i.e., a distância entre um padrão $\boldsymbol{x}$ e um neurônio $\boldsymbol{m}_i$. Este tipo de distância depende da métrica a ser usada no problema. Em geral usam-se para tal fim a distância Euclidiana ou a de Hamming. O mesmo critério de distância usado para encontrar o neurônio vencedor pode ser usada, como veremos mais adiante, para calcular distâncias entre neurônios, o que nos permitirá fazer a análise de um SOM treinado. A figura 3.9 apresenta um modo simples e funcional de visualizar o SOM. Cada neurônio possui um *'endereço'* no grid e todos recebem os mesmos sinais provenientes da entrada $\boldsymbol{x} = \{ x_1, x_2, x_3 \}$. Pode-se ver, associado a cada neurônio, um vetor na mesma dimensão da entrada (no caso *3*). Esta figura será de bom uso quando no capítulo 5 discutirmos métodos de visualização de um SOM treinado.

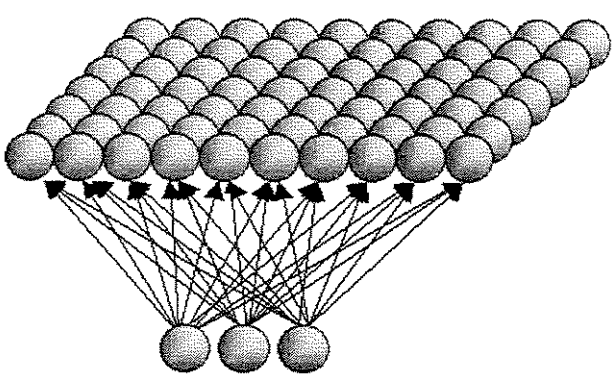


Figura 3.7 - SOM com tamanho 7x10 e dimensionalidade de entrada, p = 3.

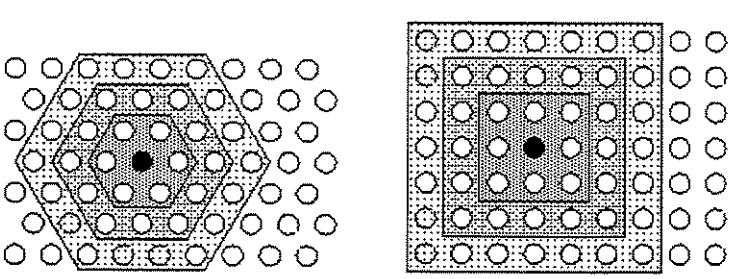


Figura 3.8 - Duas configurações de grid e níveis de vizinhança do neurônio vencedor aos neurônios circunvizinhos.

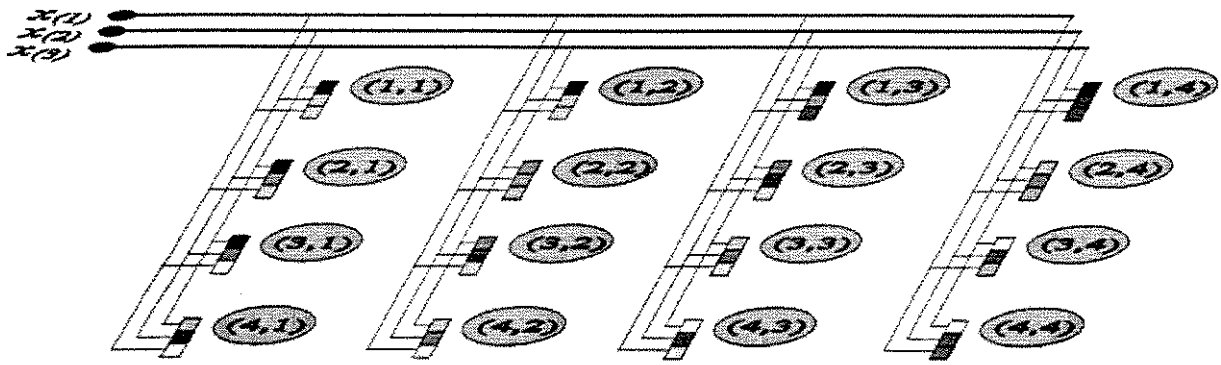


Figura 3.9 - SOM com tamanho 4x4 e dimensionalidade de entrada, p = 3.

3.5 Treinamento do SOM

3.5.1 Algoritmo convencional

Talvez uma das razões do sucesso do SOM seja a simplicidade do algoritmo de treinamento. Geralmente, o processo de treinamento do SOM consiste de apenas quatro etapas, além da inicialização:

1. Inicialização da rede. Pode-se usar estratégias descritas na seção 3.5.3 para gerar os vetores de pesos iniciais $\boldsymbol{m}_i$, $i = 1, \ldots, n$, onde n é o número de neurônios.

2. Um padrão de entrada $\boldsymbol{x}_k = (\xi_1, \xi_2, \ldots, \xi_p)$, $\boldsymbol{x}_k \in \Re^p$, é selecionado aleatoriamente de todo conjunto de padrões.

3. Uma função de ativação é usada para calcular o estado de cada neurônio i em relação ao padrão $\boldsymbol{x}_k$. No caso da distância Euclidiana, temos a equação abaixo

$$d(\boldsymbol{m}_i, \boldsymbol{x}_k) = \sqrt{\sum_{j=1}^{p} \left[x_{kj}(t) - m_{ij}(t) \right]^2} \tag{3.4}$$

4. O neurônio vencedor, c, é escolhido de acordo com a equação

$$\| \boldsymbol{x} - \boldsymbol{m}_c \| = \min_i \{ \| \boldsymbol{x} - \boldsymbol{m}_i \| \} \tag{3.5}$$

onde ||·|| denota a distância utilizada, no nosso caso a Euclidiana.

5. Os pesos sinápticos do neurônio vencedor, c, como também os pesos dos neurônios que estão dentro da vizinhança de c são atualizados através da seguinte equação

$$\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + h_{ci}(t) \cdot [\boldsymbol{x}(t) - \boldsymbol{m}_i(t)] \tag{3.6}$$

Onde t é a iteração ou passo dentro de uma época, $\boldsymbol{x}(t)$ é o padrão de treinamento, geralmente escolhido de forma aleatória do conjunto de dados no passo t, e $h_{ci}(t)$ é o núcleo de vizinhança ao redor do neurônio vencedor c no passo t. Este último termo é uma função decrescente com o tempo e com a distância do neurônio i ao neurônio vencedor c, e geralmente é formado por dois componentes: a taxa de aprendizado $\alpha(t)$ e a função de vizinhança $h(d, t)$:

$$h_{ci}(t) = \alpha(t) \cdot h(\| \boldsymbol{r}_c - \boldsymbol{r}_i \|, t) \tag{3.7}$$

Onde $\boldsymbol{r}_i$ é a posição do neurônio i na camada de Kohonen.

O processo definido pelos passos 2-5 é repetido iterativamente, o que leva a um mapeamento gradual de preservação da topologia dos sinais de entrada à medida que o algoritmo de treinamento converge.

Ao final do treinamento espera-se que o mapa esteja topologicamente ordenado. Basicamente isto significa que n_i padrões que estejam próximos no espaço p-dimensional de atributos devem ser mapeados em neurônios que estejam próximos no espaço do *grid*, geralmente no mesmo neurônio ou em neurônios vizinhos. Porém o inverso não é verdadeiro: dois neurônios vizinhos no espaço do *grid* podem estar bastante distantes no espaço de atributos. Esta informação será de grande valia quando formos segmentar automaticamente o mapa (ver capítulo 5).

3.5.2 Comentários sobre o processo de treinamento do SOM

Treinamento (ou estimação de parâmetros) em redes neurais, ou outros sistemas estatísticos, envolve geralmente a minimização (ou maximização) de um funcional no qual os parâmetros a serem estimados são grandezas independentes. No caso de treinamento supervisionado, além da entrada $x(t)$ a ser apresentada à rede neural, apresentamos a correspondente saída desejada τ. A minimização dos parâmetros geralmente usa uma

função derivada de gradiente descendente no espaço de pesos de uma função erro entre o conjunto de respostas providas pela rede $\underline{\psi}$ e o conjunto de saídas desejadas $\underline{\tau}$. O critério de término pode ser um valor pré-especificado para o funcional, um número máximo de iterações, a estabilização (no tempo) dos parâmetros, ou uma combinação destes fatores.

No caso do SOM, dado o conjunto de entradas $\Xi = \boldsymbol{x}_k$, $1 \leq k \leq n$, originários de um espaço $\Re^p$, em cada passo uma entrada $\boldsymbol{x}(t) \in \Xi$ é selecionada, geralmente de forma aleatória, para a apresentação ao SOM, garantindo uma uniformidade de apresentações de todos os $\boldsymbol{x}_k \in \Xi$. Define-se uma época como a apresentação completa do conjunto de padrões, Ξ, à rede neural.

Um dos principais problemas no treinamento do SOM é a ausência de um funcional ou um critério de otimização (ver comentário sobre tentativas de provas matemáticas do processo de convergência na seção 3.7). A ausência das informações referentes à saída desejada para cada padrão de entrada no treinamento faz com que a alternativa de adaptação dos pesos seja feita de forma não supervisionada, i.e., através de um processo competitivo. O objetivo é *´sintonizar´* os neurônios às entradas, i.e., que um neurônio vencedor (BMU[2]) de um determinado padrão de entrada responda mais intensamente àquele padrão na próxima época. O grau de adaptação, em uma iteração t, depende da taxa de aprendizado $\alpha(t)$, da função de vizinhança $h(d, t)$ e da distância entre o neurônio i, representado pelo seu vetor sináptico $\boldsymbol{m}_i$, e o vetor $\boldsymbol{x}(t)$.

A etapa do treinamento do SOM que mais consome tempo é encontrar o neurônio vencedor. Geralmente usa-se busca seqüencial, o que torna o processo extremamente custoso quando o número de neurônios é elevado. Variações do SOM têm sido propostas como a implementação de uma estrutura em árvore para tentar diminuir este tempo (Lampinen e Oja, 1992; Koikkalainen, 1994). Uma alternativa é a paralelização da busca, através de processadores especiais e/ou paralelos.

À medida que o treinamento prossegue e os padrões são apresentados à rede, a taxa de aprendizado $\alpha(t)$ decresce gradualmente a um valor pré-especificado, geralmente próximo a zero, de acordo com uma dada função de decaimento. Isto garante o término do processo de aprendizado em um tempo finito. Geralmente usa-se funções monotônicas decrescentes na faixa de valores [0, 1]. Um exemplo para $\alpha(t)$ poderia ser

$$\alpha(t) = \alpha(0) \cdot e^{-\left(\frac{t}{\lambda}\right)} \tag{3.8}$$

[2] Do inglês *best match unit.*

onde $\alpha(0) < 1$, é a taxa de aprendizado inicial e λ é um parâmetro responsável pela taxa de redução desejada.

De forma similar, a vizinhança ao redor do neurônio vencedor, $h_{ci}(t)$, equação (3.7), decresce com o tempo a uma taxa também previamente especificada. Esta vizinhança é considerada o principal agente da auto-organização no SOM (Kohonen, 1982a,b) responsável pela preservação das relações topológicas. Esta propriedade faz com que um estímulo que excita um dado neurônio η no SOM influencie, de uma certa forma, outros neurônios que estão na vizinhança de η. A influência pode ser positiva ou negativa, o que podemos denominar, respectivamente, de sinapses laterais excitatórias ou inibitórias. Esta possibilidade de adaptação conjunta de neurônios de um centro de ativação diferencia o SOM de outras redes competitivas[3] que utilizam o princípio *Winner Takes All* (WTA), como por exemplo, a *Adaptive Resonance Theory* (Carpenter e Grossberg, 1987), que só atualizam os pesos do vencedor. Pode-se definir uma região de ativação circular, ou retangular, (veja figura 3.8) como um conjunto de neurônios $N_c(t)$ em uma região do mapa centrado no vencedor, c. No início do treinamento $N_c(t)$ compreende grande parte do SOM e à medida que o aprendizado prossegue $N_c(t)$ encolhe até que apenas o neurônio vencedor seja atualizado. Por razões de simplicidade, geralmente atualiza-se apenas os pesos dos neurônios dentro da área excitatória, i.e., $h_{ci}(t) = 0$ para neurônios $\boldsymbol{m}_i \notin N_c(t)$ (ver equações 3.6-3.7).

Uma vez encontrado o BMU, o padrão $\boldsymbol{x}$ é mapeado para este neurônio. Os pesos sinápticos de BMU, assim como os pesos dos neurônios dentro da vizinhança de BMU, são deslocados na direção do padrão $\boldsymbol{x}$, de acordo com a equação 3.6. A magnitude do deslocamento é influenciada pela taxa de aprendizado $\alpha(t)$ e pela função de vizinhança $h(d, t)$. O BMU possui o vetor de pesos sinápticos com maior similaridade ao estímulo apresentado. Geralmente usa-se distância Euclidiana que eqüivale ao produto interno quando os padrões e os pesos são normalizados em $\|\cdot\| = 1$. Apesar de não obrigatória, grande parte das aplicações na literatura normalizam os dados antes de apresentarem ao SOM, limitando o espaço de entrada a uma hiperesfera p-dimensional de raio 1. A norma Euclidiana de um vetor $\boldsymbol{x} = (x_1, x_2, \ldots, x_p)^T$, pode ser definida como

$$norma = \sqrt{\sum_{k=1}^{p} x_k^2} \tag{3.9}$$

[3] Também das redes de múltiplas camadas *feedforward*, treinadas com backpropagation ou um variante deste, que atualizam todos os pesos $\boldsymbol{w}_i$ da rede, independente da escolha do vencedor.

e os padrões de entrada podem ser normalizados mudando seus componentes para

$$x'_k = x_k / norma \tag{3.10}$$

A função de ativação baseada no produto interno é descrita na equação (3.11). Todos os padrões de entrada podem ser normalizados antes da apresentação ao SOM.

$$f_i(t) = \boldsymbol{x}^T(t) \cdot \boldsymbol{m}_i(t) = \sum_{k=1}^{p} x_k(t) \cdot m_{ik}(t) \tag{3.11}$$

No caso de distância Euclidiana o BMU foi descrito como o neurônio, *c*, com menor distância (eq. 3.5). Como a equação (3.11) define uma função de similaridade entre dois vetores, o BMU será definido como o neurônio mais similar ao estímulo *x(t)*, o que levaria a equação (3.5) a ser escrita como

$$\left\| \boldsymbol{x}^T \cdot \boldsymbol{m}_c \right\| = \max_i \left\{ \left\| \boldsymbol{x}^T \cdot \boldsymbol{m}_i \right\| \right\}. \tag{3.12}$$

A equação (3.5) deve ser rescrita na forma da equação (3.12) sempre que os pesos sinápticos mantenham norma 1. Isto pode ser feito re-normalizando os pesos de todos os neurônios que devam ser atualizados (i.e., os neurônios vencedores e os que estejam dentro da respectiva vizinhança) para um dado instante de tempo *t*.

Quando os padrões e os pesos sinápticos estão normalizados, e usamos como função de ativação a distância Euclidiana, equação (2.9), pode-se provar que devido à expansão de (2.9) como apresentada na equação (3.13), buscar a menor distância entre o padrão $\boldsymbol{x}$ e os neurônios $\boldsymbol{m}_i$, i = 1, 2, ..., equivale a encontrar o máximo valor para $\boldsymbol{m}_i^T \boldsymbol{x}$.

$$\left\| \boldsymbol{x} - \boldsymbol{m}_i \right\| = (\boldsymbol{x}^T \boldsymbol{x} - 2\boldsymbol{m}_i^T \boldsymbol{x} - 1)^{1/2}. \tag{3.13}$$

3.5.3 Escolha dos parâmetros de treinamento

A escolha correta dos parâmetros do treinamento não é uma tarefa trivial. Várias heurísticas foram propostas, muitas das quais específicas para determinados tipos de problemas. O resultado final do treinamento, mesmo com parâmetros iguais, geralmente é diferente. Isto ocorre devido a basicamente dois fatores aleatórios, como a inicialização dos pesos e a

seqüência de apresentação dos padrões no treinamento. Na maioria dos casos usando o algoritmo básico apresentado na seção 3.5, também denominado algoritmo sequencial ou incremental (Kohonen, 1997a), estes fatores aleatórios levam a mapeamentos diferentes mas que possuem relações como por exemplo, rotações de um mapa em relação a outro.

3.5.3.1 Dimensionalidade e tamanho do mapa

Normalmente escolhe-se mapas de dimensão 1 ou 2, este último mais freqüente devido à capacidade de visualização do mapeamento dos dados na forma de um *display*. Entretanto, na maioria dos problemas de agrupamentos, por exemplo (Bezdek e Pal, 1992), usa-se um *grid* unidimensional, no qual o número de neurônios é igual ao número esperado de agrupamentos. No nosso caso, a menos que se explicite contrariamente, usaremos mapas de dimensionalidade 2 ou maior (capítulo 7).

O algoritmo básico (Kohonen, 1984, 1997a) fixa o número de neurônios N no início. Para problemas de agrupamentos, o ideal seria termos bem menos neurônios do que dados, de forma que a *taxa de ocupação* (t_O) ficasse em um nível razoável, onde t_O poderia ser definida como uma média do número total de padrões pelo número total de neurônios. Por exemplo, poderíamos desejar que t_O sempre fosse, no mínimo, 1. Quando temos grandes bases de dados isto se justifica, e certamente a escolha do tamanho do mapa deveria em princípio, supor $t_O >> 1$. Porém, em muitos casos práticos, pode ser útil dimensionar o mapa supondo que alguns neurônios serão inativos, i.e., não serão BMUs para nenhum padrão do conjunto de treinamento. Como veremos mais adiante, o SOM funciona como uma grade elástica tentando concentrar mais neurônios em regiões mais densas do espaço de atributos. À medida que a dimensionalidade do espaço de atributos aumenta, deveríamos alocar mais neurônios, de forma a tentar representar melhor os agrupamentos no modelo de protótipo distribuído (ver capítulo 2). Por outro lado, ainda teremos alguns neurônios que farão parte de regiões de ligação entre dois ou mais agrupamentos, que inclusive podem ser inativos. Mapas de tamanho grande tornam o aprendizado muito lento, enquanto que mapas de tamanho muito pequenos, por exemplo, 3×3, geram *U-matrizes*[4] (ver capítulo 5) também pequenas (no caso 5×5), o que irá dificultar bastante o uso de métodos de processamento e segmentação de imagens.

Pode-se ainda empregar um método dinâmico para fazer o crescimento do SOM. Uma das soluções seria aumentar o mapa inserindo novos neurônios entre os neurônios existentes. Os pesos dos novos neurônios podem ser calculados por simples interpolação linear, i.e., fazendo-se uma média dos pesos dos neurônios na vizinhança de cada neurônio inserido.

[4] Matrizes onde os elementos são distâncias entre os neurônios, calculadas no espaço de pesos.

Assim, poderíamos treinar inicialmente uma rede 5×5 para um grande número de épocas, t_1, e em seguida, aumentar a rede, por exemplo, para 9×9, inicializar os novos pesos por interpolação linear, e continuar o treinamento por mais t_2 épocas, e assim sucessivamente. Esta seria uma das várias possíveis soluções para efetuar treinamento de um mapa de tamanho grande sem que o treinamento se torne excessivamente custoso. A motivação básica deriva do fato de que o gargalo do treinamento do SOM é a busca pelo neurônio vencedor, após a apresentação de um padrão na camada de entrada. Geralmente esta busca é feita de forma seqüencial, i.e., da ordem $O(N)$, onde N é o número de neurônios da rede. Fazendo com que $t_1 > t_2 > \ldots t_n$, poderíamos obter um ganho de tempo bastante sensível, devido a que o custo computacional do treinamento para t_i épocas seria da ordem de $O(N. t_i)$ para uma rede com N neurônios, enquanto que pelo método proposto, o custo teria ordem $O(N_1.t_1 + N_2.t_2 + \ldots + N_n.t_n)$, condicionando que $N = N_n$ e que $t_i = \sum_{k=1}^{n} t_k$.

3.5.3.2 Conectividade entre os neurônios e topologia da rede

Como descrito anteriormente, no caso bidimensional, geralmente usa-se um arranjo retangular no qual os elementos são conectados aos vizinhos nos padrões de conectividade 4 ou 6, implicando na topologia retangular ou hexagonal. Os elementos que situam-se nas extremidades do mapa podem ainda ser conectados ou não. Neste trabalho, assume-se que em todos os exemplos foram usados mapas com topologias e conectividades retangulares, a menos que se explicite o contrário.

3.5.3.3 Inicialização dos pesos

Antes de iniciarmos a fase de aprendizado devemos escolher ou inicializar os pesos sinápticos. Apesar do SOM ser relativamente robusto em relação à inicialização, métodos que pudessem inicializar os pesos de forma adequada poderiam facilitar a convergência do algoritmo. A forma mais comum de inicialização é a randômica, onde atribuímos valores pequenos aleatórios aos pesos. Pode-se inicializar vetores pegando aleatoriamente padrões no conjunto de dados, e copiando estes padrões aos vetores de pesos. Uma outra forma seria pegar valores máximos e mínimos em cada dimensão e inicializar, de forma ordenada, a grade de neurônios, interpolando o espaço de acordo com o número de neurônios.

Recentemente, Kohonen defendeu a idéia da inicialização linear, sendo preferível em relação às descritas anteriormente por possibilitar irmos diretamente para a fase de convergência do algoritmo de treinamento (Kohonen, 1997a, pg. 115). Esta forma de

inicialização será usada nos experimentos desta tese, a menos que se explicite o contrário. Descreveremos, a seguir, de forma bastante abreviada, os conceitos deste método.

A inicialização linear de um mapa com espaço de saída com dimensão k é feita utilizando-se dos k componentes principais da matriz de autocorrelação do conjunto de dados X. Considere o caso de um SOM bidimensional ($k = 2$). Autovalores e autovetores são definidos por $\boldsymbol{A} \cdot y = e \cdot y$, onde $\boldsymbol{A}$ é a matriz de autocorrelação de dados, y é um autovetor e e o autovalor correspondente.

Os pesos, $\boldsymbol{m}_i(0)$, são inicializados de forma ordenada na direção do sub-espaço linear obtido pelos autovetores, ortogonais, correspondentes aos dois maiores autovalores, sendo seu centróide coincidente com a média do conjunto de dados, X. Seja y_1 e y_2 os autovetores escolhidos. As coordenadas do neurônio (i, j) podem ser expressas por

$$s \cdot \left[\left(i - \frac{\max\_i}{2} \right) \cdot y_1 + \left(j - \frac{\max\_j}{2} \right) \cdot y_2 \right]$$

onde max_i e max_j representam o tamanho do mapa bidimensional, escolhido pelo usuário, e s é uma constante selecionada de forma adequada.

3.5.3.4 Funções para decaimento da vizinhança ao redor do BMU e da taxa de aprendizado

A função de vizinhança $h_{ci}(t)$, dada pela equação (3.7), tem uma importância fundamental no SOM, atuando como um núcleo de suavização sobre a grade de neurônios. Para convergência do algoritmo, é necessário que $h_{ci}(t) \rightarrow 0$ quando $t \rightarrow \infty$. A função $h_{ci}(t) \approx h(\| r_c - r_i \|, t)$, onde r_c e $r_i \in \Re^2$, são as posições dos neurônios c e i no *grid* de neurônios, respectivamente. À medida que $\| r_c - r_i \|$ aumenta, $h_{ci}(t) \rightarrow 0$.

O algoritmo original do SOM (Kohonen, 1982a) define uma vizinhança do tipo bolha ou circular, i.e., dado um raio $r(t)$ do neurônio vencedor, no passo t, todos os neurônios dentro deste raio possuem h igual a 1. Seja $N_c(t)$ o conjunto de neurônios satisfazendo esta condição. Desta forma, $h_{ci}(t) = \alpha(t)$ para todos os neurônios $i \in N_c(t)$ e $h_{ci}(t) = 0$ caso contrário ($i \notin N_c(t)$). A figura 3.7 ilustra o efeito no tempo de $N_c(t)$.

Outra forma bastante usada é vizinhança do tipo Gaussiana,

$$h_{ci}(t) = \alpha(t) \cdot \exp\left(-\frac{\| r_c - r_i \|^2}{2 \cdot \sigma^2(t)} \right) \tag{3.14}$$

onde σ pode ser definido como o raio de $N_c(t)$. Deve-se sempre escolher raio inicial $N_c(0)$ relativamente grande, por exemplo, a metade do diâmetro da rede. Escolhendo $N_c(0)$ pequeno pode fazer com que o algoritmo não gere uma ordenação adequada, o que Erwin *et al.* (1992a,b) denominaram meta-estados.

Como descrito na equação (3.7), $h_{ci}(t)$ também é proporcional a $\alpha(t)$, e ambos, componentes de $h_{ci}(t)$, $h(\| r_c - r_i \|, t)$ e $\alpha(t)$, devem ser escolhidas como funções monotonicamente decrescentes com o tempo, t. Em relação a $\alpha(t)$ duas condições são suficientes e necessárias (aproximação estocástica de Robbins & Monro (1951)):

$$\sum_{t=1}^{\infty} \alpha^2(t) < \infty \quad \text{e} \quad \sum_{t=1}^{\infty} \alpha(t) = \infty. \tag{3.15}$$

A escolha de $\alpha(t)$ deve ser feita de forma criteriosa principalmente em mapas de grande dimensão. De acordo com (3.15), vemos que $\alpha(t)$ é limitada, e inicialmente deveríamos escolher $\alpha(t)$ próximo a 1, possibilitando seu decaimento com o tempo, que pode ser feito, por exemplo, de forma exponencial, linear, ou inversamente proporcional a t. O algoritmo em lote (seção 3.5.4) elimina o problema de escolher uma seqüência ótima para $\alpha(t)$.

3.5.3.5 Número de épocas

Como o treinamento é um processo estocástico, a precisão do mapeamento obtido depende do número de épocas. O algoritmo original (Kohonen, 1982a) estabelece duas fases, uma de ordenação inicial, e outra de convergência. A primeira duraria, por exemplo, cerca de 1000 épocas, enquanto a convergência deveria ser mais demorada, por exemplo, 100.000 épocas.

Em relação à apresentação de dados à rede, passo 2, o algoritmo convencional seleciona padrões aleatoriamente do conjunto de dados durante uma época. Porém, caso existam padrões raros em uma base de dados, e seja importante obter uma representação deles no SOM, podemos usar esquemas que dêem ênfase a tais padrões, forçando reapresentações do mesmo objeto numa mesma época de treinamento. Isto pode ser feito modificando a

probabilidade de ocorrência do padrão $\boldsymbol{x}$, $p(\boldsymbol{x})$, justificado apenas quando têm-se um bom conhecimento *a priori* do conjunto $\boldsymbol{X}$.

Provas de convergência foram tentadas por várias abordagens, incluindo processos de Markov (Bouton et al., 1991; Cottrell et al., 1994), e por equações diferenciais ordinárias (Flanagan, 1994, 1996). Porém apenas no caso unidimensional, e para um sinal de entrada unidimensional, foi possível demonstrar o processo de ordenação do SOM (ver Kohonen(1997a), Cottrell (1997). Em alguns casos, avanços foram efetuados porém em algoritmos com simplificações em relação ao SOM básico (Flanagan, 1994). Uma análise matematicamente rigorosa da dinâmica do algoritmo de treinamento e prova de convergência de um SOM com *grid* bidimensional, ou que tenha *grid* unidimensional porém possua mais de uma entrada, ainda não foram publicadas. Porém, condições gerais para obter bons mapas incluiam treinar a rede com número de épocas relativamente elevado, e na fase inicial manter tanto o raio de vizinhança do neurônio vencedor como a taxa de aprendizado relativamente elevados, permitindo decaimento suave destes fatores com o tempo.

3.5.4 Algoritmo em lote ou paralelo

Um método de tornar o resultado do mapeamento insensível à seqüência de apresentações de padrões à rede é deixar para atualizar os pesos apenas no final de uma época. Isto pode ser feito somando-se a média das contribuições de todos os padrões para cada neurônio. Teremos então um deslocamento médio que na prática tem conduzido a melhores resultados que o algoritmo convencional. Em relação ao algoritmo apresentado na seção 3.5.1, a equação 3.6 teria no termo mais à direita um somatório na forma como apresentada na equação 3.16

$$\boldsymbol{m}_i(t_e+1) = \boldsymbol{m}_i(t_e) + \frac{1}{n}\sum_{l=1}^{n}\left\{h_{ci}(t_l)\cdot\left[\boldsymbol{x}(t_l) - \boldsymbol{m}_i(t_l)\right]\right\} \tag{3.16}$$

Onde t_e e t_l são iterações das épocas e dos passos dentro das épocas, respectivamente, $\boldsymbol{x}(t_l)$ é o padrão de treinamento geralmente escolhido de forma aleatória do conjunto de dados no tempo t_l e $h_{ci}(t_l)$ é o núcleo de vizinhança ao redor do neurônio vencedor c no tempo t_l, e n é o número de padrões usados no treinamento. Geralmente escolhe-se um valor pequeno e fixo para a taxa de aprendizado $\alpha(t)$, componente de $h_{ci}(t_l)$, como por exemplo, 0.05.

A implementação prática deste algoritmo pode ser feita de forma simples alocando uma lista de tamanho n para cada neurônio. Cada iteração pode ser usada a equação (3.6) e a quantidade de deslocamento para cada neurônio $\boldsymbol{m}_i$, i.e., $\Delta \boldsymbol{m}_i(t_l)$, em cada iteração t_l pode ser acumulada nesta lista, na posição l. No final de cada época soma-se as contribuições para cada neurônio e limpa-se a lista. Por outro lado, pode-se ter uma variável associada a cada neurônio, que seria a simplificação de uma lista de tamanho 1, na qual as contribuições são agregadas ao longo das iterações dentro de uma época. A vantagem do uso de uma lista de tamanho n é que podemos extrair alguma estatística a mais dos deslocamentos dos pesos no espaço dentro de uma época e em algumas épocas consecutivas. Por exemplo, poderíamos extrair informações dos deslocamentos de cada neurônio para cada padrão nas últimas três épocas para saber a trajetória média que o neurônio está realizando no espaço e usar esta informação para acelerar a convergência do algoritmo. A desvantagem do algoritmo em lote é a necessidade de se dispor de todos os dados antes do início do treinamento.

3.5.5 Um algoritmo mais eficiente para redução do tempo de treinamento

Uma maneira de diminuir o tempo de treinamento seria diminuir o tamanho do raio da busca pelo neurônio vencedor a um dado padrão. Como esta busca é feita de forma seqüencial, como descrito na seção 3.5.2, ela é uma das operações que mais consomem tempo no treinamento do SOM. Usando a propriedade de que, à medida que o treinamento avança os centros de ativação especializados em padrões vão surgindo, podemos supor que quanto mais treinamos um SOM, maior será a probabilidade de que o vencedor, c, para um padrão $\boldsymbol{x}$ em uma dada época t_e seja o mesmo vencedor na época anterior t_{e-1}, ou que esteja nas proximidades de c. Assim, podemos fazer com que cada padrão $\boldsymbol{x}$ tenha uma variável associada ao último neurônio vencedor, $c(\boldsymbol{x})$, na época anterior, e efetuar a busca pelo vencedor na época atual apenas em uma sub-região do mapa delimitada por um raio de busca que decresce com o tempo, $R(t_e, c)$. Pode-se usar qualquer formato de função monotônica decrescente para $R(t_e, c)$ porém devemos preferir funções suaves e que possibilitem no final do treinamento um raio de busca superior à vizinhança final $h(d, t)$. Assim, teríamos inicialmente R igual ao tamanho do mapa, e poderíamos fazê-lo decrescer à medida que o número de épocas aumentasse, porém sempre mantendo $R(t_e, c) > h(d, t)$. O uso deste algoritmo será de grande importância principalmente à medida que usemos mapas de maior tamanho e dimensão, como no caso do mapa com *grid p-dimensional* (capítulo 7). O número total de neurônios pode ser bastante elevado, por exemplo, um mapa com *grid* de dimensão *4* contendo *10×10×10×10* neurônios possui 10^4 neurônios ao todo. Para cada iteração de cada época significa buscar o vencedor c entre 10^4 neurônios. Decrescendo este espaço de busca, por exemplo, em uma dada época, t_e, centrado no vencedor do padrão $\boldsymbol{x}$ na

última época, $c(t_{e-1}, \boldsymbol{x})$, poderíamos ter um raio, por exemplo, de tamanho 5, o que daria $5 \times 5 \times 5 \times 5 = 625$ neurônios, contra 10^4, ou seja, uma redução para 6.25% do espaço de busca original. Esta redução de tempo integrada para todas as iterações e para todas as épocas resulta em uma grande redução do tempo total do treinamento.

3.6. Exemplo de uso do SOM

Esta seção apresenta um exemplo simples do uso do SOM para o caso de entrada bidimensional. A figura 3.10 ilustra um conjunto de dados gerado por uma mistura de três Gaussianas bivariadas, com vetor de médias $\mu = \{ \mu_1, \mu_2, \mu_3 \} = \{ (0, 0), (1, 0), (0.5, 0.866) \}$. O número total de objetos é 375, sendo 125 objetos por classe. A matriz de covariâncias, para as três classes é $\sigma^2 \cdot I$, onde σ usado foi 0.15.

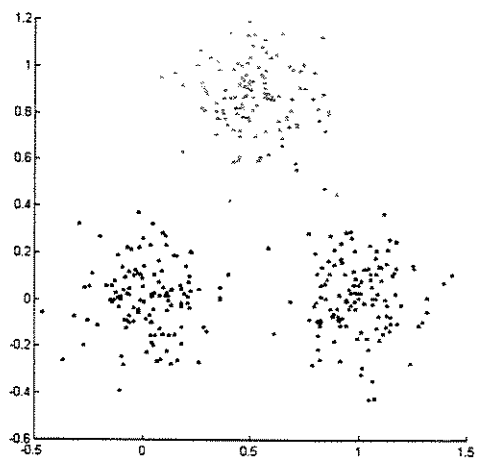


Figura 3.10: Conjunto de dados gerado artificialmente

A figura 3.11 ilustra os contornos das funções componentes da misturas de Gaussianas obtidas após simulação do algoritmo *Expectation-Maximization* (EM) ver seção 2.7), supondo conhecido o número de agrupamentos, 3. Os vetores de médias obtido foram, μ_1 = (0.0311, 0.0049), μ_2 = (0.9966, 0.0125), e μ_3 = (0.5187, 0.8638). As matrizes de covariâncias obtidas foram

$$\Sigma_1 = \begin{bmatrix} 0.0252 & 0.0005 \\ 0.0005 & 0.0212 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 0.0223 & -0.0001 \\ -0.0001 & 0.0262 \end{bmatrix}, \text{e } \Sigma_3 = \begin{bmatrix} 0.0251 & -0.0015 \\ -0.0015 & 0.0200 \end{bmatrix},$$

ambas as três muito próximas à matriz Σ usada para gerar os dados, $\Sigma = \begin{bmatrix} 0.0225 & 0 \\ 0 & 0.0225 \end{bmatrix}$.

Note que, pelo fato do modelo de dados ser originalmente Gaussiano, e por apresentar médias relativamente distantes entre si, o EM conseguiu recuperar com boa precisão a estrutura dos dados. Os pesos dos componentes obtidos também foi muito próximo do ideal (0.3333), $\pi_1 = 0.3341$, $\pi_2 = 0.3375$ e $\pi_3 = 0.3284$. Diferenças devem-se ao número reduzido de dados disponíveis (número finito).

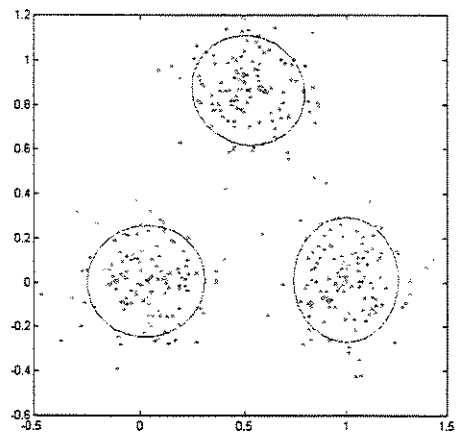


Figura 3.11: Contornos dos componentes da mistura de Gaussianas obtido pelo algoritmo EM, K = 3.

As densidades são mostradas nas figuras 3.12 e 3.13.

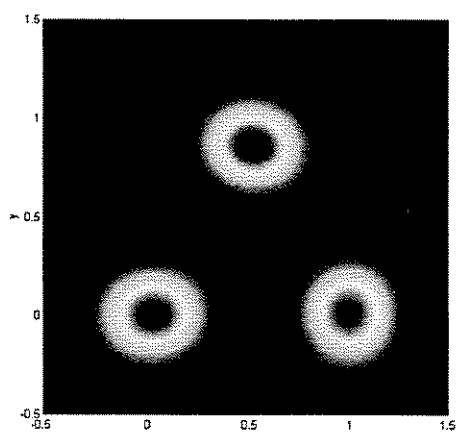


Figura 3.12: Densidades dos componentes da mistura de Gaussianas obtidos pelo algoritmo EM (em 2D).

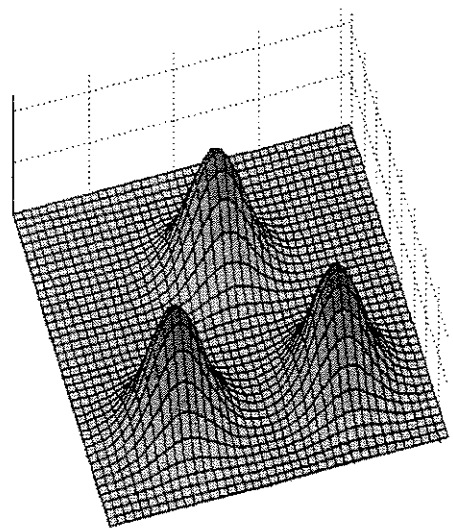


Figura 3.13: Densidades dos componentes da mistura de Gaussianas obtidos pelo algoritmo EM (em 3D).

Usando *k-means*, e também supondo $K = 3$, as figuras 3.14 e 3.15 ilustram o resultado obtido após convergência.

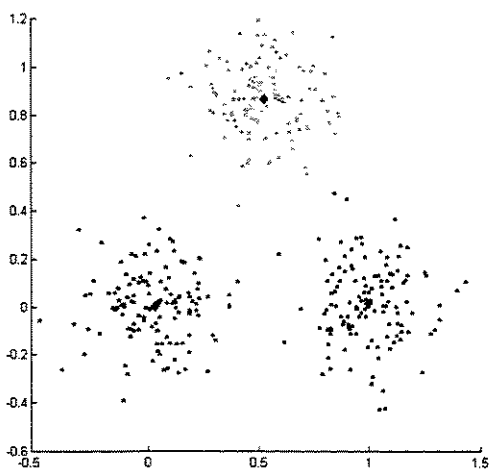


Figura 3.14: Resultado da classificação para k-means, K = 3.

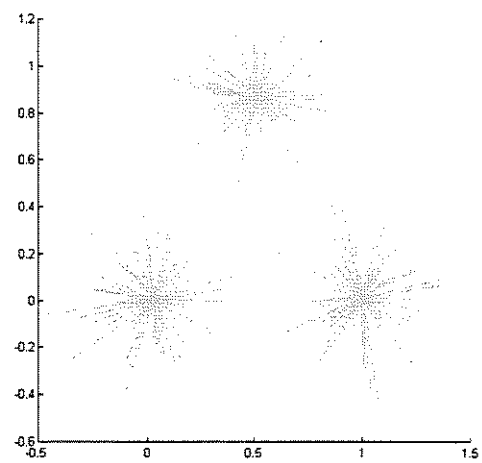


Figura 3.15:Ligações dos objetos às médias mais próximas, para o resultado apresentado na fig. 3.13.

Um dos problemas do k-means, e também da abordagem por misturas de Gaussianas, é que caso escolhamos errado o valor de K, os métodos, que necessitam deste valor no processo de otimização, tentarão impor K agrupamentos aos dados. Por exemplo, uma simulação no caso de misturas de Gaussianas com $K = 6$ é apresentado nas figuras 3.16 e 3.17.

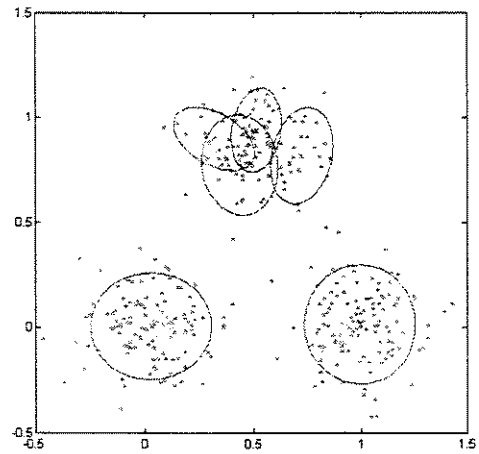


Figura 3.16: Contornos dos componentes da mistura de Gaussianas obtidos pelo algoritmo EM, K = 6.

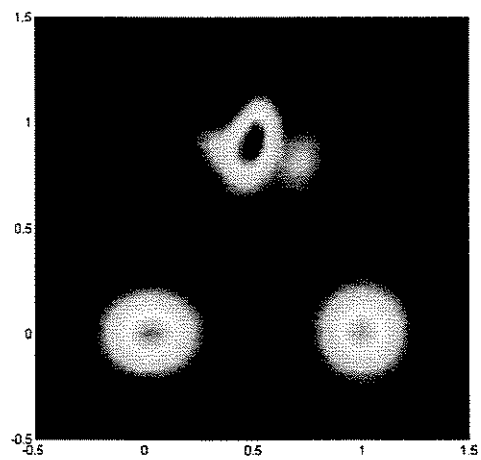


Figura 3.17: Densidades dos componentes da mistura de Gaussianas obtidos pelo algoritmo EM (em 2D), K = 6.

A figura 3.18-a ilustra *grid* de um mapa unidimensional com 40 neurônios após inicialização linear utilizando os dados apresentados na figura 3.10. Após 1000 iterações do algoritmo em lote, temos o *grid* como mostrado na figura 3.18-b, na qual podemos visualizar como a estrutura concentrou neurônios em regiões de maior densidade de pontos. No capítulo 7 discutimos um pouco sobre a implicação do mapeamento de um espaço de entrada de dimensão maior que a dimensão do *grid* do SOM. A figura 3.18-c ilustra o histograma de vencedores, H, ou seja, quantas vezes o neurônio i foi vencedor após o mapa estar ordenado. Por exemplo, o neurônio 3 teve 15 padrões associados a ele.

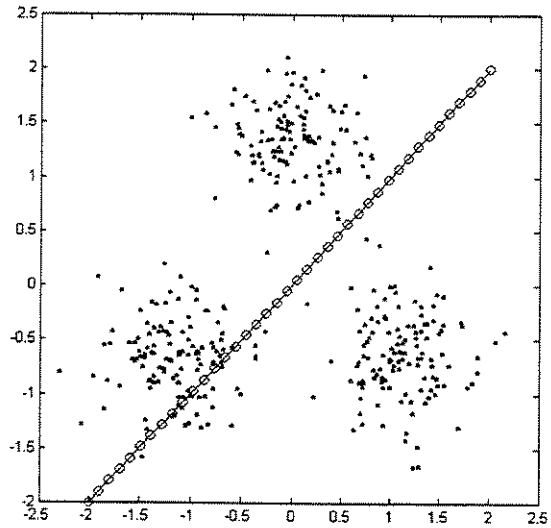


Figura 3.18-a: Grid do SOM 40x1 após inicialização linear.

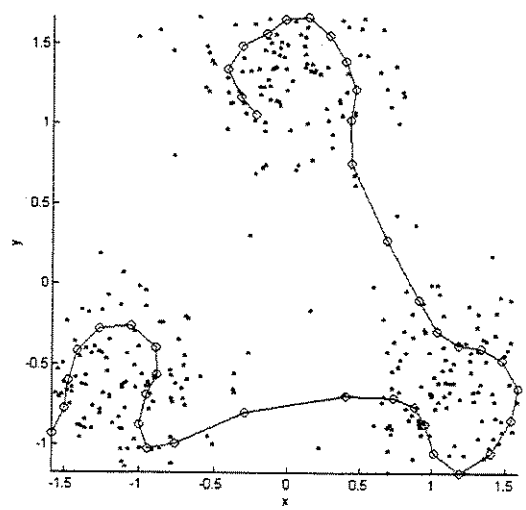


Figura 3.18-b: Grid do SOM 40x1 após 1000 iterações batch.

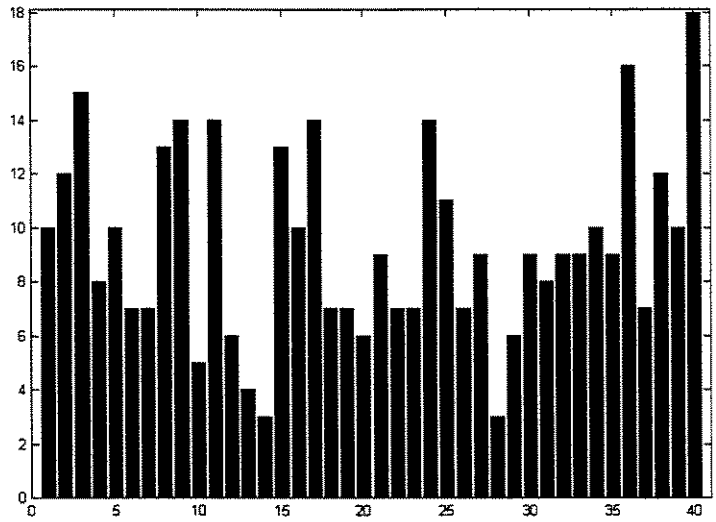


Figura 3.18-c: Histograma de vencedores para o SOM 40x1

Para o caso de um SOM bidimensional, escolhemos o tamanho 10×10 para o mapa e o inicializamos de forma linear. O resultado é apresentado na figura 3.19. Após 1000 iterações do algoritmo em lote (*batch*) obteve-se a configuração mostrada na figura 3.20. Novamente, note que houve uma concentração de neurônios nas regiões onde há maior densidade de objetos.

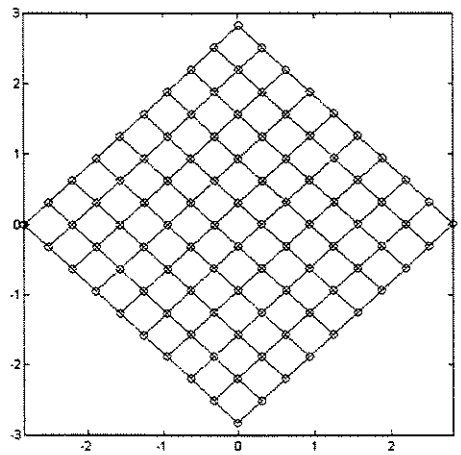


Figura 3.19: Grid após inicialização linear de uma rede SOM de tamanho 10x10.

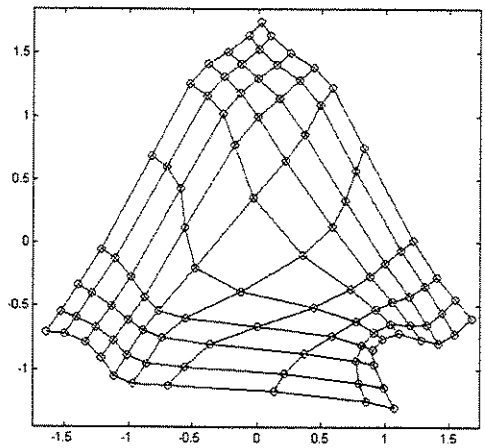


Figura 3.20: Grid do SOM 10x10 após 1000 iterações do algoritmo em lote e inicialização linear.

O histograma de vencedores, H, neste caso é também bidimensional e está apresentado na forma tabular na figura 3.21, representando quantas vezes o neurônio i foi vencedor após o mapa estar ordenado. Por exemplo, o neurônio (1, 1) obteve 6 padrões na sua área de quantização, ou de influência, enquanto que o neurônio (10, 10) obteve 10 padrões. Alguns neurônios obtiveram 0 vencimentos. Alguns autores chamam estes neurônios de células mortas (Zupan & Gasteiger, 1993). À medida que aumentamos o número de neurônios em uma rede, constata-se um aumento de neurônios mortos. Pode ser conveniente definir grau de ativação como um valor mínimo, φ, que os neurônios deveriam possuir para serem considerados para análise. Neurônios inativos podem ser definidos como neurônios cujo número de vencimentos seja inferior a φ, i.e., se $H(i, j) \leq \varphi$. Geralmente é mais conveniente trabalhar com percentuais para φ do que com valores como os apresentados na figura 3.21.

6	3	0	3	5	8	3	8	6	8
3	3	1	2	3	2	2	3	5	5
2	4	1	2	5	6	9	3	6	7
7	1	3	0	2	6	1	5	3	4
1	2	4	0	1	2	0	0	0	1
4	6	0	5	1	1	3	5	6	8
1	3	5	5	1	4	2	9	3	5
12	5	3	3	2	3	2	3	8	5
3	1	6	3	0	4	4	8	3	4
5	5	5	2	1	6	7	2	6	10

Figura 3.21: Histograma de vencedores para o SOM 10x10

A figura 3.22 ilustra a quantização obtida pelo SOM. Por razões de claridade da imagem, as linhas simbolizando vizinhança entre os neurônios, mostrada na figura 3.20 foram eliminadas. Os elementos ilustrados pelo símbolo (+) são os neurônios, enquanto que os dados são os pontos (•). Cada objeto é ligado ao neurônio mais próximo (vencedor) por uma linha. Note a existência de neurônios mortos, i.e., $H(i, j) = 0$. Geralmente tais neurônios aparecem em regiões com baixa densidade de probabilidade de objetos, sendo sua existência decorrente da estrutura da grade elástica do SOM.

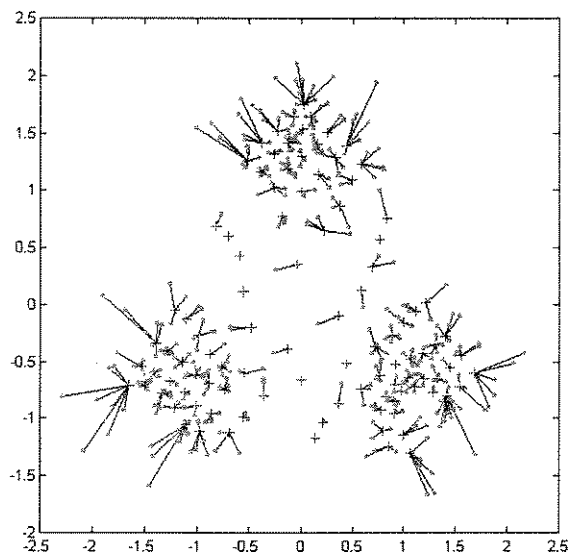


Figura 3.22: Quantização obtida após 1000 iterações.

Uma maneira de pensar como ocorre a quantização no SOM é através do diagrama de Voronoï. Cada neurônio possui uma região de influência (ou célula de Voronoï) no espaço, na qual todos os pontos são mais próximos a tal neurônio do que a qualquer outro. Uma forma de visualizar a classificação de cada padrão em cada neurônio é através da superfície de influência dos neurônios. O objetivo é visualizar a configuração de um mapa com dimensão dos dados de entrada 2D, onde a distância de um ponto qualquer do espaço ao neurônio mais próximo é plotada no eixo z. As figuras 3.23 e 3.24 ilustram a superfície de influência, em 2D e 3D, respectivamente. Pode-se limitar a influência de cada neurônio a um valor máximo, ρ, de forma similar ao parâmetro de vigilância da rede ART. Por exemplo, fazendo $\rho = 0.5$ para todos os neurônios, obtemos uma figura como a mostrada em 3.25. Este mecanismo pode, por exemplo, prevenir o sistema contra valores discrepantes (*outliers*).

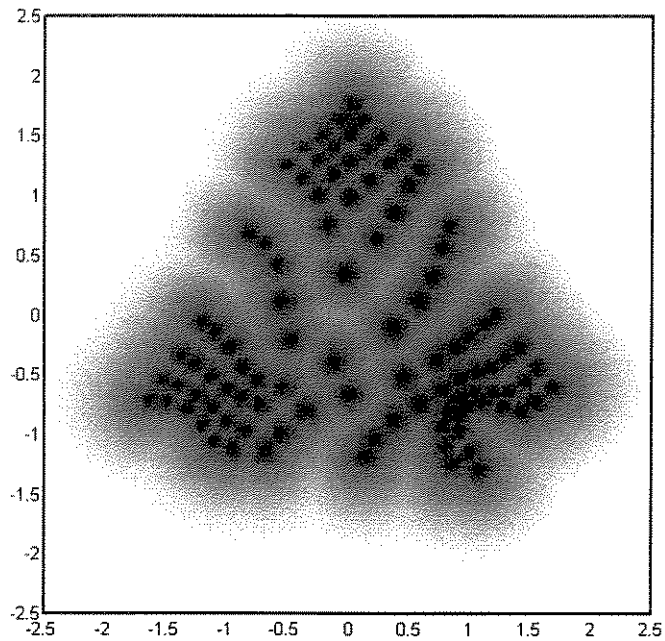


Figura 3.23: Superfície de influências dos neurônios em 2D

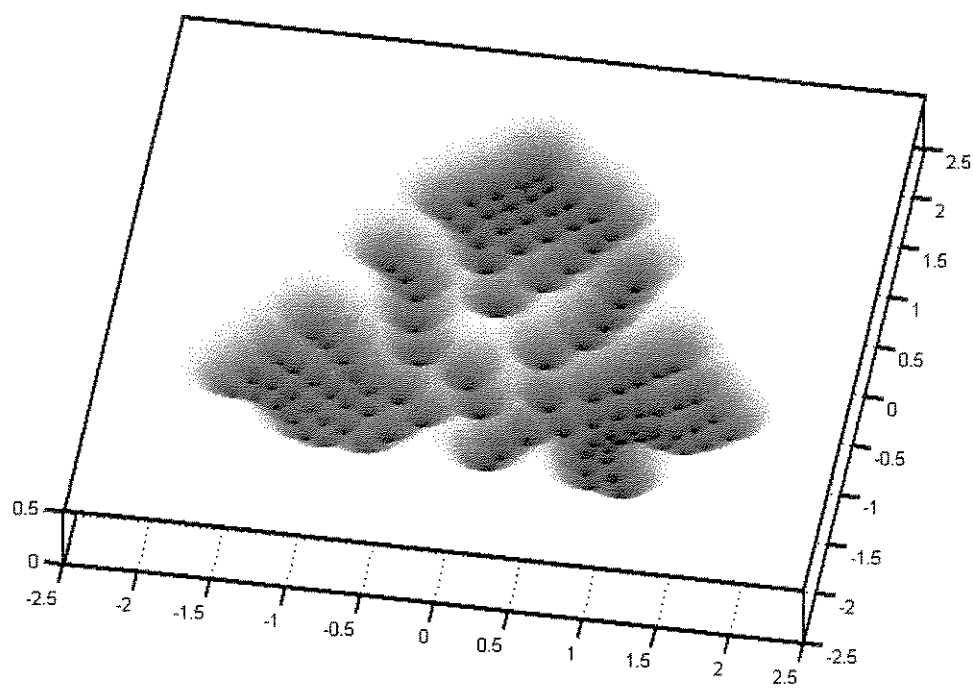


Figura 3.24: Superfície de influências dos neurônios em 3D, limitada em 0.5

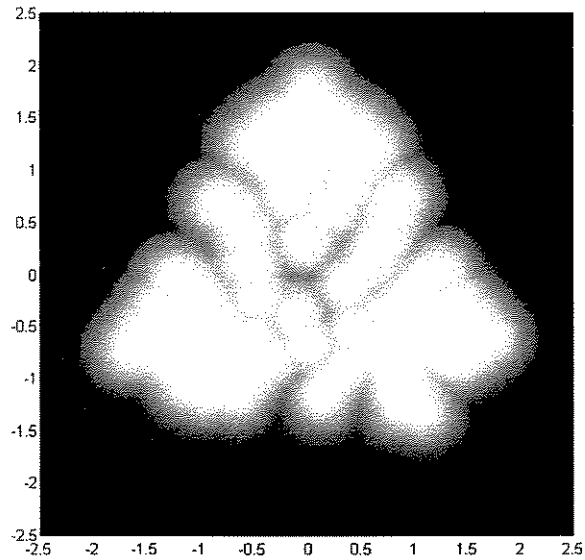


Figura 3.25: Superfície de influências dos neurônios em 2D, limitada em 0.5

Apesar de todos os neurônios serem importantes, a configuração obtida após o treinamento pode ser melhor analisada caso olhemos a superfície de influência dos neurônios que realmente estão ativos, para um dado φ escolhido. A figura 3.26 ilustra a quantização caso desconsiderássemos neurônios cujo $H(i, j)$ seja menor ou igual a 1. A figura 3.27 ilustra a superfície de influências para este caso. Note na figura 3.28, que é uma visualização da superfície de influências para neurônios com $H(i, j) > 1$, e limitada em $\rho = 0.5$, que já é possível perceber a existência de três grandes centros de atração, separados por uma borda, que na figura 3.27 é ilustrada pelas regiões mais claras da imagem.

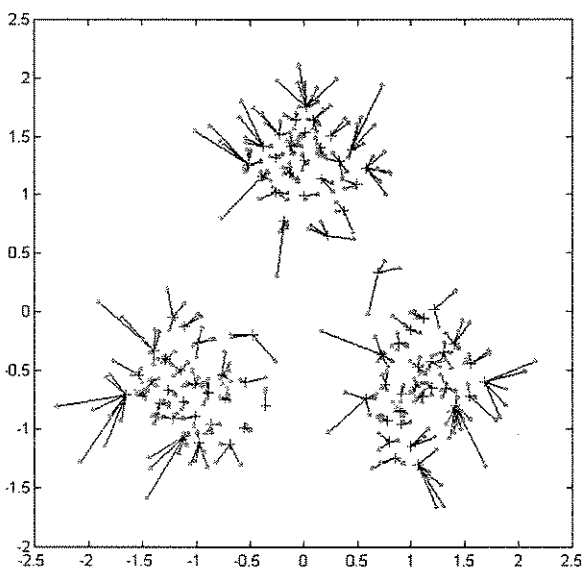


Figura 3.26: Quantização obtida considerando apenas neurônios ativos com H(i, j) > 1.

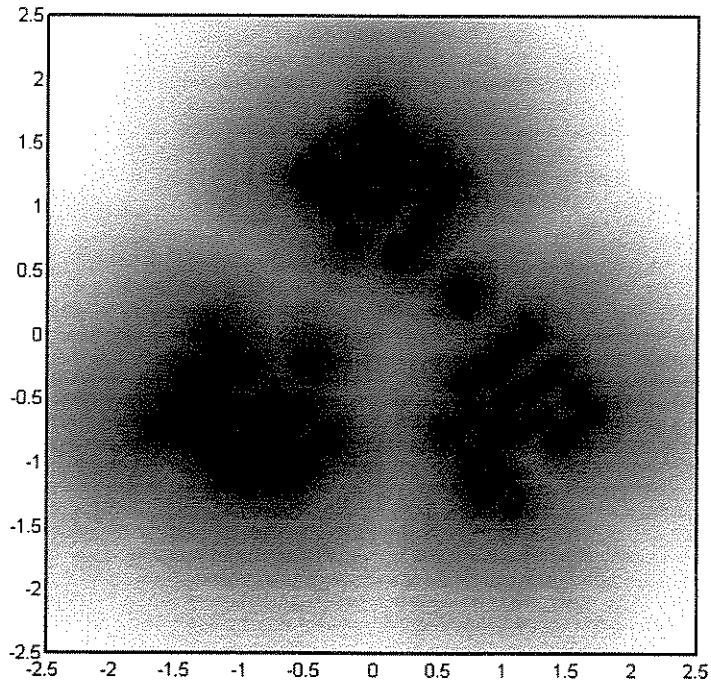


Figura 3.27: Superfície de influências para a configuração de neurônios apresentado na figura 3.26, H(i, j) > 1.

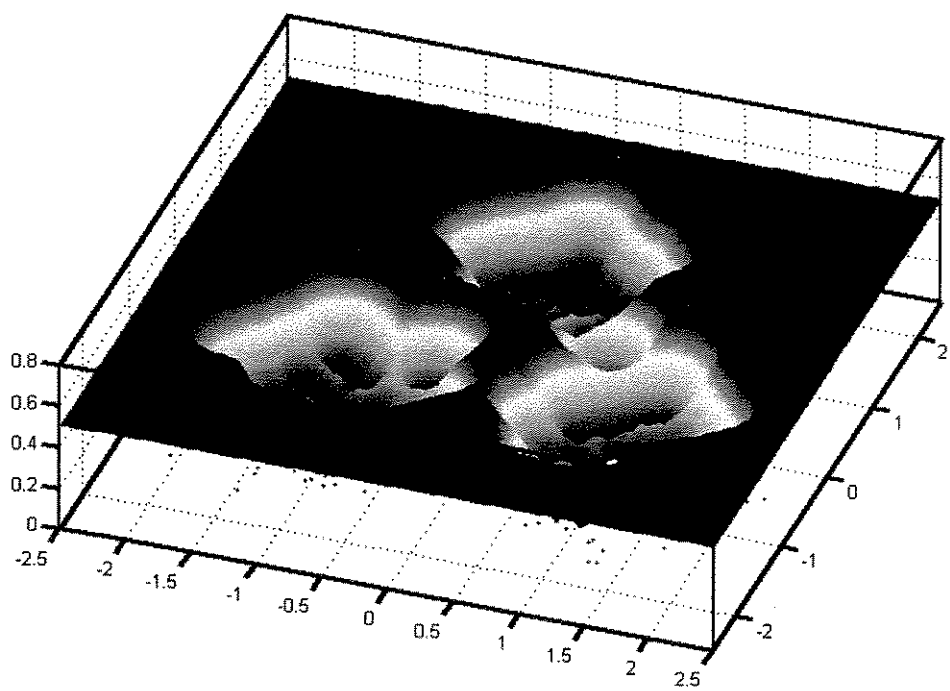


Figura 3.28: Superfície de influências apresentada na figura 3.25, H(i, j) > 1, limitada em 0.5.

Sendo ainda mais rigoroso, i.e., aumentando φ para 3, obtemos as figuras apresentadas em 3.29 - 3.31. Fica evidente a existência de três grandes centros de atração (ver fig. 3.31).

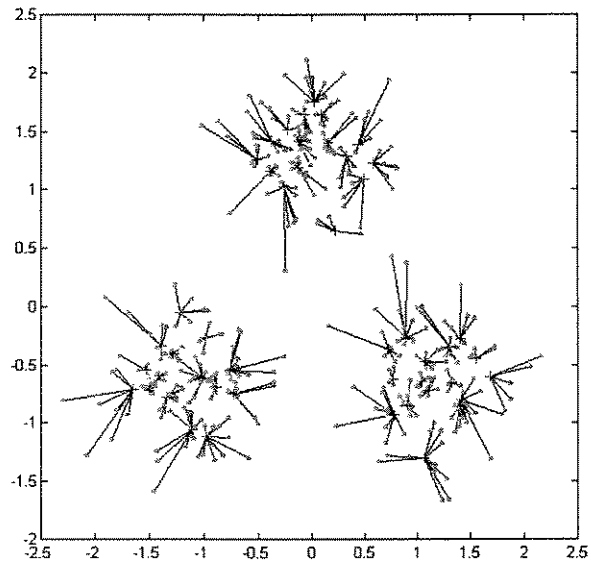


Figura 3.29: Quantização obtida considerando apenas neurônios ativos com H(i, j) > 3.

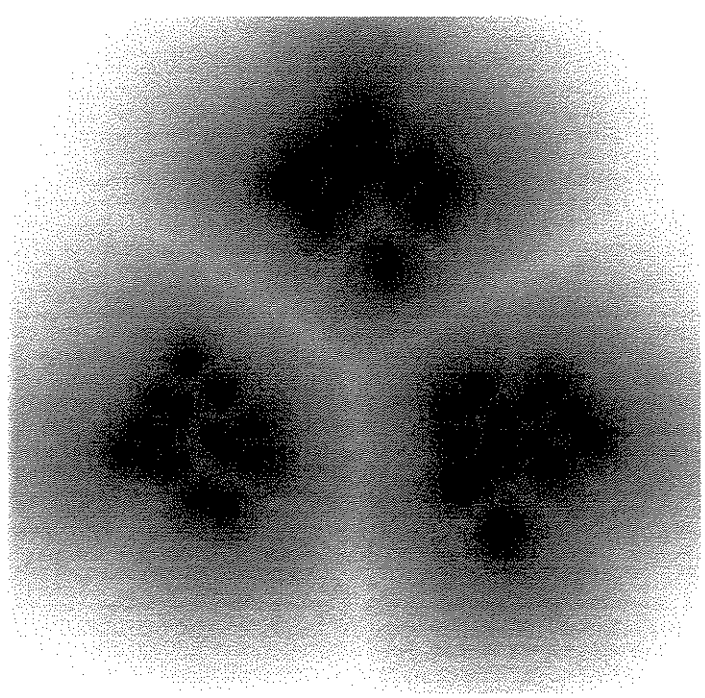


Figura 3.30: Superfície de influências para a configuração de neurônios apresentado na figura 3.29,H(i, j) > 3.

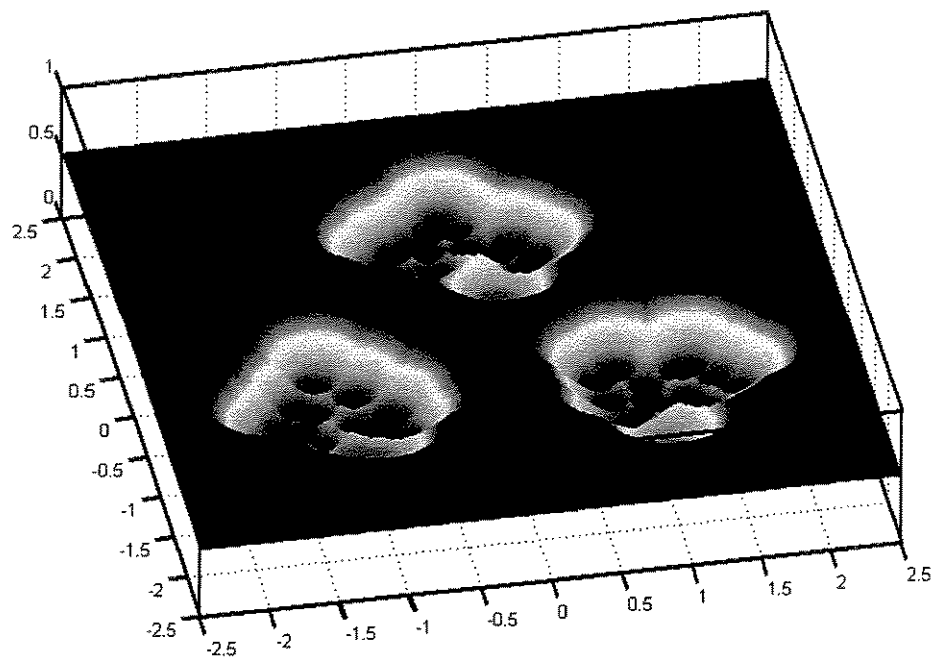


Figura 3.31: Superfície de influências para a configuração de neurônios apresentado na figura 3.29, H(i, j) > 3, limitada em 0.5.

Aumentando ainda mais φ para 7, obtemos as figuras 3.32 - 3.36. Vê-se que à medida que aumentamos φ, descobrimos centros de atração que correspondem a agrupamentos de neurônios mais fortes, pois retiramos neurônios inativos. Porém, deve-se manter em mente que a quantização do SOM não prevê exclusão de neurônios inativos, e este artifício pode ser usado, como será demonstrado posteriormente, para facilitar a segmentação da grade de neurônios.

Infelizmente a superfície de influências dos neurônios (ou dos protótipos), assim como o diagrama de Voronoï, não é de muita utilidade quando a dimensão do espaço de entrada é maior que 2, sendo seu uso mais recomendado para entendermos o processo de quantização efetuado pelo SOM.

Outros métodos serão mostrados nos capítulos posteriores (por exemplo a U-matrix, capítulo 5, e o mapa de ativações dos neurônios, capítulo 6) que possibilitam a descoberta de agrupamentos em problemas com dimensões do espaço de entrada arbitrárias.

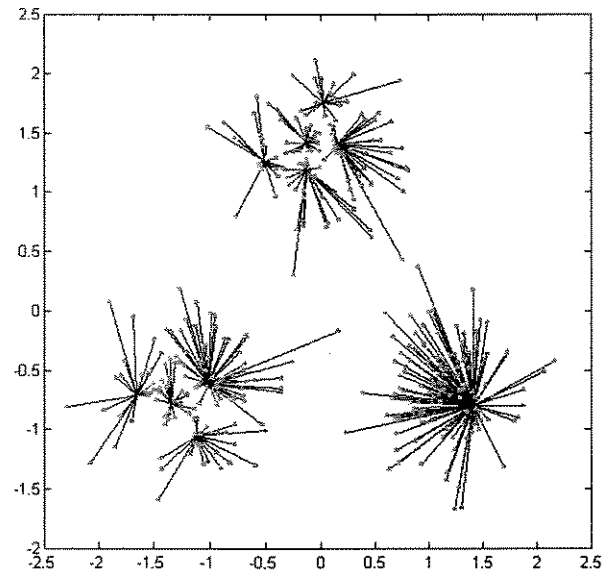


Figura 3.32: Quantização obtida considerando apenas neurônios ativos com H(i, j) > 7.

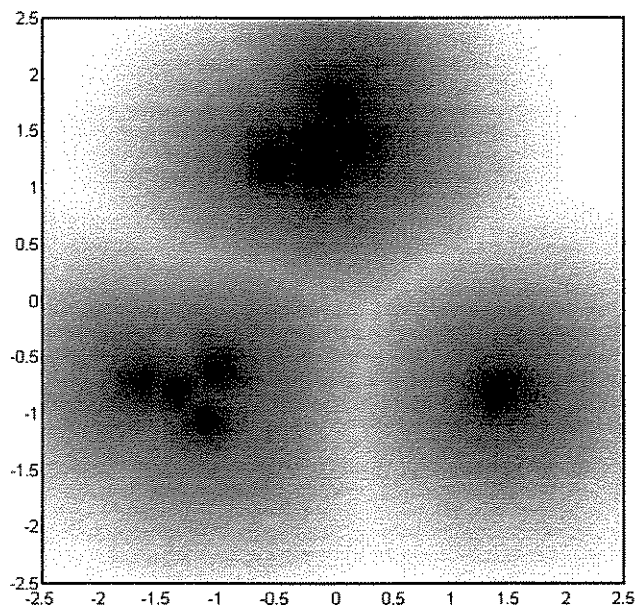


Figura 3.33: Superfície de influências para a configuração de neurônios apresentado na figura 3.32,H(i, j) > 7

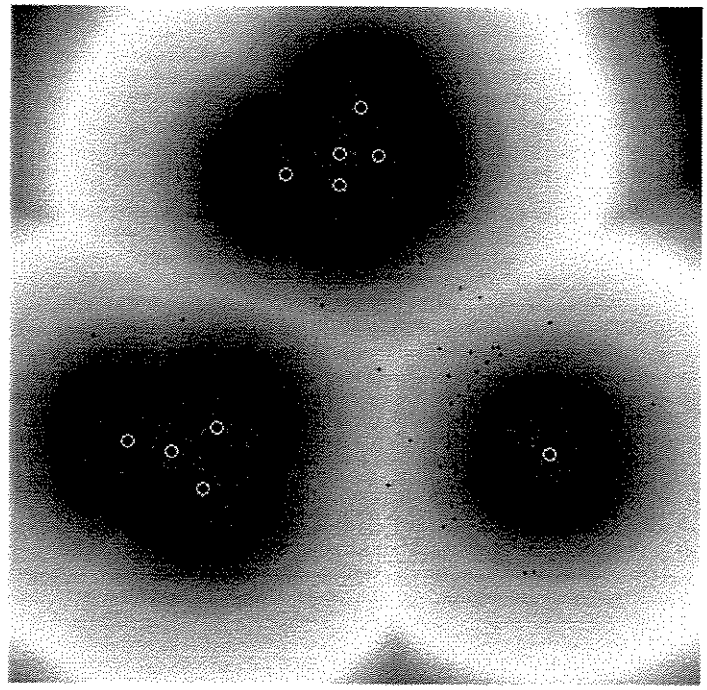


Figura 3.34: Superfície de influências para a configuração de neurônios apresentado na figura 3.32,H(i, j) > 7, mostrando os neurônios e os objetos.

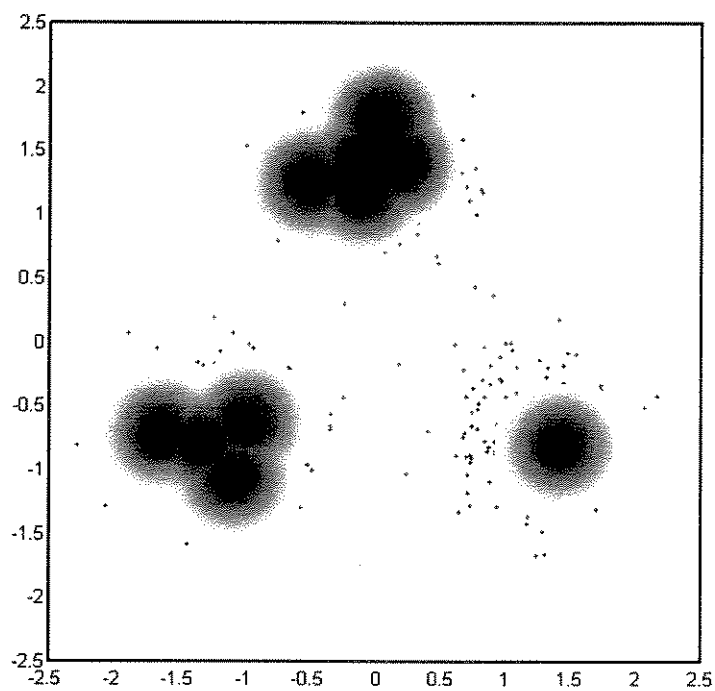


Figura 3.35: Superfície de influências para a configuração de neurônios apresentado na figura 3.32, H(i, j) > 7, mostrando os neurônios e os objetos, com área de influência dos neurônios, ρ limitado a 0.5.

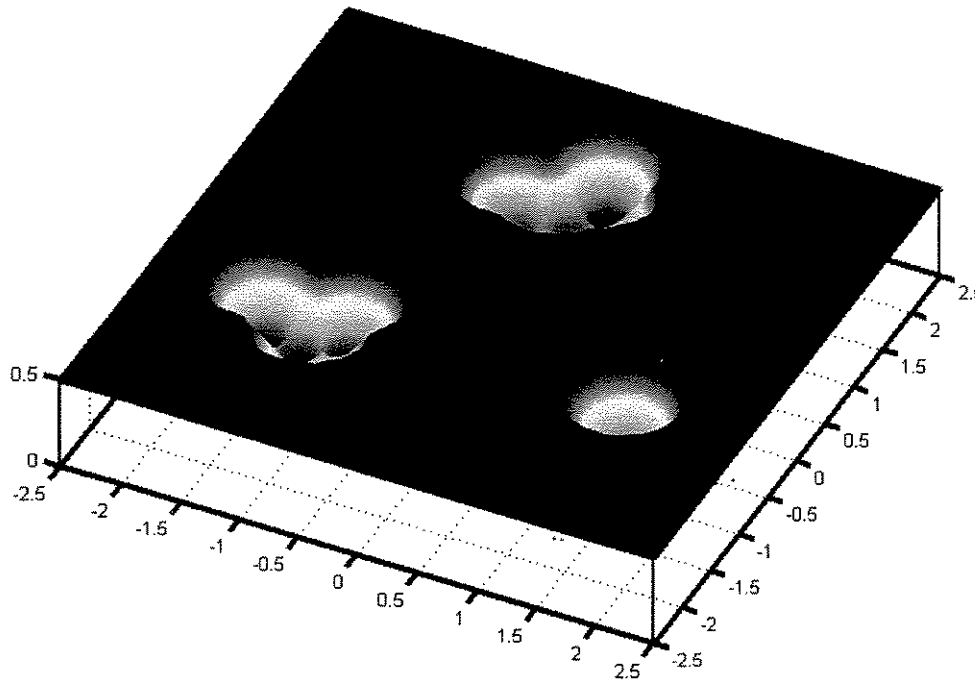


Figura 3.36: Superfície de influências apresentada na figura 3.32, H(i, j) > 7, limitada em 0.5.

3.7. Aplicações e literatura do SOM

Atualmente, o SOM é o modelo de rede neural mais importante no paradigma de redes neurais competitivas. São inúmeras as aplicações atuais do SOM, e da mesma forma que ocorre com análise de agrupamentos e classificação automática (ver capítulo 2) atualmente há disponível uma imensa quantidade de artigos e relatórios publicados nas mais variadas áreas, de engenharia a medicina, de biologia a economia, etc. O livro de Kohonen (1997a) traz mais de 2300 referências. Há disponível na internet uma bibliografia da área contendo mais de 3300 referências (Kaski *et al.*, 1998). A seguir, fazemos uma resenha da literatura do SOM, destacando algumas das publicações importantes no desenvolvimento da área. Uma análise mais completa pode ser vista em Kohonen (1997a).

Atualmente praticamente todos os livros na área de redes neurais dedicam pelo menos um capítulo ao SOM, porém livros com maior ênfase para o SOM incluem: Kohonen (1989a; 1997a), Ritter *et al.* (1992), Zupan & Gasteiger (1993). Artigos de revisão incluem:

Kohonen (1990), Oja (1995), Kohonen *et al.* (1996c), Ritter & Schulten (1988), Allinson (1992), Cottrell (1997) e Cottrell *et al.* (1998), este último com forte apelo matemático.

Apesar do SOM ser um algoritmo relativamente simples, tanto seu código quanto sua simulação, suas propriedades teóricas permanecem ainda sem provas, no caso geral, apesar de grande esforço da comunidade científica no sentido de fechar esta questão. Atualmente, apenas modelos de SOM unidimensionais (rede linear), e com entrada unidimensional, foram completamente analisados. Vários autores propuseram métodos para tentar provar aspectos matemáticos da organização efetuada pelo SOM. O primeiro a tentar provar o processo de ordenação foi Kohonen (1982b, 1989a). Vários autores consideraram cadeias de Markov, incluindo Erwin et al. (1992a,b), Bouton e Pagès (1994), e Benaïn et al. (1997). Houve tentativas de definir funções de energia ou critérios que pudessem ser minimizados de forma a garantir um treinamento eficiente (Tolat, 1990; Heskes e Kappen, 1993; Erwin et al., 1992b), sendo que estes últimos estudaram também a importância da função de vizinhança no processo de treinamento. Antes disso, Ritter et al. (1986, 1988) descreveram sobre um estado estacionário em qualquer dimensão arbitrária, porém o estudo se restringiu à fase final de ordenação, após a auto-organização, e não provaram a existência deste estado estacionário. Não é possível definir no espaço multidimensional uma boa configuração ordenada de neurônios, que seja estável para o algoritmo e que possa ser considerada um estado de absorção. Não existe em um espaço multidimensional uma noção de ordem (Cottrell et al., 1995; Fort e Pagès, 1996). Ainda, no SOM bidimensional a cadeia de Markov é irredutível, o que não acontece no caso unidimensional (Cottrell et al., 1998). Por outro lado, Erwin et al. (1992a,b) demonstraram ser impossível associar uma função potencial decrescente, e global, para o algoritmo, pelo menos quando a distribuição de probabilidade dos padrões é contínua. Interpretação do mecanismo de organização do SOM relacionando com modelos fisiológicos foi apresentado em Kohonen (1993). Flanagan (1994, 1996) propôs a análise do SOM através de sistemas de equações diferenciais, porém várias simplificações foram feitas no modelo básico do SOM para viabilizar a análise.

Tentativas de acelerar a convergência abordaram o uso de um termo *momentum*, similar ao utilizado no algoritmo *backpropagation* (Hagiwara, 1996), usando filtros de Kalman (Yin e Allinson, 1993) e com o uso de lógica nebulosa (Pal et al., 1992, 1993). Apesar de terem denominado *fuzzy Kohonen clustering networks*, Bezdek et al. (1992) usam a mesma estrutura básica do SOM e existe lógica nebulosa apenas na taxa de aprendizado que foi modificada com o objetivo de aumentar a velocidade do treinamento.

Modelos construtivos foram propostos principalmente para permitir ao sistema descobrir automaticamente uma estrutura que represente o espaço p-dimensional da melhor maneira

possível. Exemplos incluem o modelo *Growing Cell Structures* (Fritzke, 1993, 1994) e o *Incremental Grid Growing* (Blackmore e Miikkulainen, 1993, 1995), entre outros, que serão mais detalhados no capítulo 4. Outras extensões incluem a proposta de hierarquizar mapas, que inicialmente foram propostas para permitir treinamento mais rápido (Koikkalainen, 1994), e posteriormente para habilitar uma flexibilização de agrupamentos obtidos (Miikkulainen 1993; Costa e Netto, 1999c). Outros modelos variantes do SOM incluem Sirosh e Miikkulainen (1993) que desenvolveram sistemas inspirados em modelos biológicos do córtex visual, e Kangas et al. (1990) que apresentaram pequenas variações em relação ao modelo básico (Kohonen, 1990). Modelos híbridos do SOM com outras arquiteturas de redes neurais, especialmente a ART (*adaptive resonance theory*) incluem Baraldi e Parmiggiani (1995) e Hetch-Nielsen (1987).

Aplicações industriais foram citadas em vários artigos, como por exemplo, controle de processos e telecomunicações (Kohonen *et al.*, 1996c; Visa, 1992). Aplicações como compressão e codificação de imagens foram abordadas por vários autores, sendo uma das principais aplicações na área de processamento de imagens. Exemplos incluem Burel e Pottier (1991), Lu e Shin (1992), Corral et al. (1994) e Kangas e Kohonen (1996). Em geral, a codificação foi aplicada usando pequenos blocos da imagem para treinamento, o que permitiu a geração de um conjunto de protótipos que são usados na quantização de imagens para transmissão ou armazenamento. Uma excelente revisão de métodos de codificação, incluindo abordagens neurais como o SOM foi apresentado em Dony e Haykin (1995). Segmentação de imagens e texturas foi abordado por Oja (1991, 1992), Visa (1990a, b; 1994), Wan e Fraser (1993, 1994a), e Lancini (1994). Análises de imagens médicas foram apresentados, por exemplo, em Pan e Chen (1992) e Turner et al. (1993). Uma das maiores aplicações do SOM têm sido em reconhecimento de fonemas e de voz. Kohonen (1984, 1988, 1989b, 1990) apresentou a máquina fonética (*phonetic typewriter*) a qual baseia-se em um SOM bidimensional treinado com partes do espectro de sinais de voz, obtidos pela transformada de Fourier, onde houve um ajuste fino, i.e., calibração do mapa, para aumentar a taxa de reconhecimento. O sistema é capaz de reconhecimento de fala usando trajetórias de ativações entre os neurônios que são ativados pelos diferentes fonemas. Uma das aplicações vislumbradas é tradução automática, por exemplo, em telecomunicações (Kohonen, 1992).

Em automação várias aplicações robóticas, incluindo controle de trajetória de manipuladores e braços robóticos foram apresentadas, por exemplo, Ritter et al. (1989). O uso de visão computacional em controle foi abordado, por exemplo em (Ritter et al., 1992), Walter e Schulten (1993), Hesselroth et al. (1993), e Jones e Vernon (1994). Várias outras aplicações como controle de trajetórias e navegação foram descritos, incluindo Kröse e Eecen (1994) e Ball (1994). Controle de processos químicos, detecção e identificação de

seqüências de proteínas, caracterização e quantificação de sistemas micro-biológicos, etc., foram abordados por vários autores, incluindo Ferrán e Ferrara (1992), Gasteiger e Zupan (1993), Zupan e Gasteiger (1993), Goodacre (1994), Merelo et al. (1994) e Tokutaka et al. (1999). Estas e outras várias aplicações, como em projetos de circuitos elétricos (Mitchison, 1995), aplicações médicas e farmacêuticas (Weinstein et al., 1995), economia e administração (Serrano et al., 1993, Wilson, 1994, Kaski e Kohonen, 1996), recuperação e armazenamento de informações (Scholtes, 1991), pesquisa operacional e otimização (Favata e Walker, 1991), pesquisa neurofisiológica e neurociências (Obermayer e Blasdel, 1997), entre tantas outras.

Várias aplicações têm sido relatadas tanto em pesquisas quanto em produtos já disponíveis nas áreas de mineração de dados e descoberta de conhecimento em bases de dados[5]. A motivação básica é a necessidade de analisar, de forma não supervisionada, grandes volumes de registros em bancos de dados, onde cada registro é composto por vários campos ou variáveis, e busca-se detectar estruturas nos dados e/ou entender os relacionamentos destes. Mapas neurais, como o SOM, têm sido utilizados, porém, na maioria dos casos, muito do sucesso da aplicação dos métodos deve-se à experiência e/ou conhecimento *a priori* dos mapeamentos a serem interpretados. O uso do SOM como ferramenta de mineração de dados (*data mining*) tem sido abordada tanto em pesquisa como em produtos tais como o WEBSOM (Kohonen, 1997b, 1998) e Clementine (1998).

O WEBSOM foi desenvolvido com o objetivo de organização automática de grandes bases de dados de textos, principalmente os disponíveis na internet, como os disponíveis em listas de grupos de interesse, enquanto que o Clementine é uma ferramenta para bancos de dados de uso mais geral. Porém ambos os aplicativos efetuam agrupamentos nos dados a partir de visualização, i.e., há necessidade de intervenção do usuário que guia manualmente a escolha dos parâmetros e a segmentação da rede. Kohonen (1997a,b) descreve o uso de uma rede com 100.000 neurônios aplicada para agrupamentos de mais de um milhão de documentos relativos a grupos de interesses da internet, totalizando cerca de 250 milhões de palavras. O processo de busca de documentos baseia-se em agrupamentos de assinaturas, que são seqüências de três palavras no texto. O resultado pode ser acessado *on line* no endereço http://websom.hut.fi/websom.

Flexer (1999) após várias simulações usando conjuntos de dados com estrutura conhecida, descreve que a ferramenta Clementine sempre escolhe número de agrupamentos inadequados. Em ambos os casos, Clementine e WEBSOM, o SOM foi utilizado apenas como um instrumento de visualização para indicar tendências de agrupamentos, sendo a partição efetuada manualmente.

Várias arquiteturas especiais foram projetadas para otimizar tanto o treinamento quanto a execução do SOM ou de um de seus modelo derivados, incluindo computadores vetoriais e paralelos, além de circuitos integrados específicos. Exemplos incluem König et al. (1993) que usaram processadores vetoriais, Whittington e Spracklen (1994) e Wyler (1993) que usaram uma plataforma multi-processada, Siemon e Ultsch (1990) que usaram transputers, e Ienne e Viredaz (1994) que apresentaram circuitos integrados digitais para acelerar o treinamento e adaptação de pesos em hardware.

Esta seção teve o objetivo apenas de mostrar quão expressivo tem sido o SOM nas mais variadas áreas da ciência, e seria praticamente impossível, na atualidade, condensar em poucas linhas o que está disponível. Um adendo desta tese é um sistema de banco de dados relacional contendo cerca de 4000 referências, grande parte delas relacionadas ao SOM, e muitas delas com resumos e comentários, que pode ser consultada por várias formas, por autor, assunto, título, ano, palavras chaves, etc., além de possuir outros arquivos como o de autores, com resumos biográficos, e links para suas páginas na *internet*.

3.8. Sumário

Este capítulo descreveu a ferramenta básica que iremos explorar para detecção de agrupamentos de geometria qualquer, o SOM. Discutimos aspectos do treinamento e seus parâmetros, e uma forma de interpretação de como é o processo de quantização efetuado após o treinamento. Por fim, destacamos que a melhor alternativa atual no treinamento é a conjunção de inicialização linear e o uso do algoritmo em lote. O primeiro fator implica em começar o treinamento em um estágio relativamente avançado de ordenação em relação à inicialização convencional, que é a aleatória. O algoritmo em lote além de ser mais rápido é insensível à ordem de apresentação dos dados, o que não ocorre no algoritmo convencional (seqüencial). Além disto, usando o algoritmo em lote não temos a preocupação da escolha da taxa de aprendizado, $\alpha(t)$, a qual pode inclusive ser fixada em um valor pequeno e constante, ex. 0.05. Em capítulos seguintes voltaremos a discutir algumas propriedades do SOM, estendendo análises e formas de visualização dos mapas no problema apresentado na seção 3.6 e em outros bancos de dados. Mostraremos, no capítulo 5, como segmentar automaticamente um mapa treinado.

[5] Do termo em inglês *Data Mining & Knowledge Discovery in Databases.*

Capítulo 4

Redes neurais competitivas e modelos derivados do SOM

Este capítulo objetiva uma breve descrição de alguns modelos de redes neurais competitivas, incluindo variantes do SOM, com o objetivo de classificação automática de dados. Algumas das idéias apresentadas aqui serão estendidas ou adaptadas aos sistemas propostos nos capítulos posteriores.

4.1 Redes neurais competitivas

Em vários problemas reais não dispomos de dados completos do mapeamento entrada / saída para treinamento da rede. Nos casos onde existem apenas os padrões de entrada, deve-se encontrar uma forma de extrair as informações relevantes, ou a estrutura natural presente nos dados, a partir das próprias amostras de treinamento. Alguns exemplos dos problemas incluem: (*i*) agrupamento de dados, onde subgrupos menos heterogêneos devem ser encontrados, caso existam; (*ii*) quantização vetorial, onde um conjunto reduzido de vetores referência deve representar um conjunto de dados. Aplicações são inúmeras, por exemplo em processamento de sinais e imagens e em telecomunicações; (*iii*) redução de dimensionalidade. Geralmente dados apresentados em grande número de dimensões podem ser submetidos a uma composição de variáveis, com o objetivo de reduzir a dimensionalidade onerando o quão menos possível o processo de determinação das características discriminatórias dos dados. Seria interessante que um sistema inteligente aprendesse tal mapeamento de forma que maximize a preservação da variância dos dados originais; (*iv*) extração de atributos, onde o sistema deveria ser capaz de detectar e extrair características de um sinal de entrada. Este problema está relacionado ao anterior (redução da dimensionalidade).

Algumas abordagens neuro-computacionais foram propostas para resolver problemas como os descritos. O modelo mais básico de aprendizado competitivo foi proposto por Rumelhart e Zipser (1985). Dentre os modelos de redes neurais auto-organizadas mais conhecidos, destacam-se o *Self-Organizing Map* (Kohonen, 1982a, 1997a), descrito no capítulo 3, o *Neocognitron* de Fukushima (1988), a *Adaptive Ressonance Theory* de Carpenter e

Grossberg (1987), e outras como a arquitetura de ligações dinâmicas de von der Maldsburg (Wiskott & von der Malsburg, 1996). A seguir, faz-se um breve comentário sobre estes modelos.

4.1.1 Adaptive Resonance Theory (ART)

A família de modelos ART (*Adaptive Resonance Theory*) foram propostos por S. Grossberg (1976) e G. Carpenter (Carpenter & Grossberg, 1987, 1988, 1991), com objetivo de descobrir agrupamentos no conjunto de padrões de forma não supervisionada. Este tipo de rede neural implementa, de forma complexa, a idéia do algoritmo convencional de agrupamentos *leader-follower* (Späth, 1980a), que é extremamente sensível à seqüência de apresentação dos dados e ao parâmetro que controla o raio máximo aceitável de um agrupamento existente. Geralmente associada à inspiração de modelos biológicos, a ART usa aprendizado competitivo e provê uma solução ao dilema da *estabilidade-plasticidade*, i.e., como aprender novos padrões sem alterar o aprendizado já efetuado. A motivação básica seria treinamento contínuo, por exemplo, em um ambiente onde as condições alteram-se ao longo do tempo. ARTs geram agrupamentos a partir de uma seqüência arbitrária de padrões de entrada e são descritas matematicamente por equações diferenciais não lineares.

Redes ARTs baseiam-se no princípio de que nossas percepções são confrontadas com nossas expectativas (modelo interno dos objetos) e que o reconhecimento de objetos é dependente da forma com que os objetos percebidos foram categorizados (Grossberg, 1995). Quando a rede é inicializada, um número máximo de agrupamentos é definido como sendo o número de neurônios disponível na camada de saída. Inicialmente não há nenhum agrupamento (i.e., neurônio ativo, representando um agrupamento). O aprendizado ocorre comparando-se os estímulos da entrada com modelos internos, pesos sinápticos que representam agrupamentos. Caso a distância do padrão ao conjunto de pesos mais próximo exceda um limiar (parâmetro de vigilância), um novo agrupamento é formado, caso existam neurônios ainda não atribuídos a nenhum agrupamento. Caso a distância esteja dentro do limiar a algum agrupamento existente, os pesos sinápticos correspondentes são adaptados. Esta é uma das razões do fato da rede ART exibir um alto grau de plasticidade, no momento de geração de um novo agrupamento, e não afetar os dados já aprendidos, i.e., os outros agrupamentos (Zurada, 1992). Ao contrário do SOM, apenas o padrão de maior resposta é associado ao padrão a ser treinado, de forma a caracterizar-se uma estrutura neural do tipo '*grandmother cell*' (ver seção 4.1.2).

O termo "ressonância" decorre do estado "ressonante" de aprendizado da rede, que emprega *loops* de realimentação entre a camada de saída e a camada de entrada, e que aprende apenas durante este estado ressonante. Vários modelos são descritos na literatura. No mais simples, ART1 apenas padrões binários são permitidos. ART2 é similar ao ART1 com a possibilidade de usar padrões analógicos, e ART3 apresenta princípios organizacionais (incorpora no modelo 'transmissores químicos') que permitem o processo de busca em uma estrutura hierárquica (Carpenter & Grossberg, 1991). Fuzzy ART é uma modificação do modelo ART para suportar lógica nebulosa. O modelo ARTMAP é uma versão supervisionada que pode aprender mapeamentos arbitrários de padrões binários. Uma versão nebulosa deste último modelo é o Fuzzy ARTMAP que também possui mecanismo de aprendizado supervisionado. Uma breve descrição do funcionamento do ART1, baseado em Beale & Jackson (1990), é apresentado a seguir.

A rede ART1 consiste de duas camadas de neurônios totalmente conectadas, a camada de entrada e a camada de reconhecimento, possuindo dois conjuntos de vetores de pesos, um '*top-down*' e um '*bottom-up*', representados, respectivamente, por $\boldsymbol{W}$ e $\boldsymbol{T}$. Em cada camada existem ainda sinais de controle (*C1* e *C2*) para direcionar o fluxo de dados nas camadas durante cada estágio do ciclo de operação. Quando *C1* está ativo (nível lógico 1) um padrão inserido na camada de entrada é passado, através de $\boldsymbol{W}$, para a camada de reconhecimento. Caso algum neurônio da camada de reconhecimento esteja ativo, *C1* é forçado para o nível lógico 0. O outro sinal de controle, *C2*, habilita ou desabilita os neurônios na camada de reconhecimento. *C2* será 1para qualquer padrão de entrada válido, e será zero quando haja falha no teste do parâmetro de vigilância.

Entre as camadas de entrada e de saída, há também um circuito de *reset*, utilizado quando deve-se alocar um novo neurônio a uma nova classe de padrões.

As fases de operação da ART1 podem ser divididas nas fases de inicialização, reconhecimento, comparação e busca. A inicialização dos pesos da matriz de realimentação, $\boldsymbol{T}$, é feita de forma simples: todos os valores serão 1, o que faz com que todos os neurônios da camada de saída estejam inicialmente conectadas a cada neurônio da camada de entrada. Os pesos da camada de alimentação, $\boldsymbol{W}$, são inicializados por

$$w_i = \frac{1}{1+n_i}$$

onde n_i é o número de neurônios na camada de entrada. O parâmetro de vigilância é inicializado na faixa $0 < \rho < 1$.

Os neurônios na camada de entrada possuem três entradas: um componente do vetor de entrada $\boldsymbol{x}$, o sinal de realimentação da camada de saída, e o sinal *C1*. O fluxo de sinais da canada de entrada é controlado pela regra "dois-terços" sugerida por Carpenter & Grossberg (1988): caso duas entradas em um neurônio estiverem ativas então a saída é 1, caso contrário o neurônio tem saída zero.

Na fase de reconhecimento, um padrão de entrada $\boldsymbol{x}$ é inserido na camada de entrada e calcula-se uma distância entre $\boldsymbol{x}$ e todos os pesos da camada $\boldsymbol{W}$. Um neurônio vencedor é encontrado, de forma similar ao SOM. Caso vários neurônios respondam para uma dada entrada, a inibição lateral entre neurônios da camada de saída faz com que, após alguns passos, apenas um neurônio possua ativação máxima, enquanto os outros estarão com ativação nula.

O neurônio vencedor passa sua classe de volta para a camada de entrada através dos pesos armazenados em $\boldsymbol{T}$, também denominada camada de comparação. A regra dos "dois-terços" é aplicada para calcular a saída de cada neurônio. Efetua-se um AND lógico entre o padrão de entrada e o sinal de realimentação da camada de saída (classe), produzindo um novo vetor (Z) na saída da camada de comparação. Z é passado para o circuito de *reset* juntamente com o vetor de entrada $\boldsymbol{x}$. Tal circuito é responsável por testar a similaridade entre $\boldsymbol{x}$ e Z, e ainda comparar com o parâmetro de vigilância. Este teste pode ser calculado de forma simples, efetuando-se o produto interno entre $\boldsymbol{x}$ e Z, e dividindo o valor pelo número de 1's no vetor de entrada.

$$S = \frac{\sum t_{ij} x_i}{\sum x_i}$$

A razão, S, é então comparada com o parâmetro de vigilância. Se $S > \rho$ a classificação está completa e o neurônio ativo indica a classe do padrão $\boldsymbol{x}$. Caso contrário, a rede não encontrou a classe correta e deve-se passar para a fase de busca, na tentativa de encontrar um novo vetor na camada de saída para o padrão de entrada.

O neurônio ativo é desabilitado. O padrão de entrada é novamente aplicado à camada de entrada, e as ativações na camada de reconhecimento são recalculadas, desconsiderando-se os neurônios desabilitados. Uma nova fase de comparação inicia-se e efetua-se um novo teste com a nova classe selecionada e o parâmetro de vigilância. O processo é repetido, desabilitando consecutivamente neurônios na camada de saída, até que um neurônio seja encontrado o qual produza um resultado satisfatório, i.e., o casamento entre o padrão de entrada e o vetor representando a classe do padrão esteja dentro dos limites do parâmetro de

vigilância. Caso não seja encontrado nenhum neurônio que satisfaça tal condição, a rede toma a decisão de gerar uma nova classe para o padrão de entrada, alocando um neurônio da camada de saída ainda não utilizado. A seguir apresenta-se brevemente o algoritmo da rede ART1, adaptado de Beale & Jackson (1990).

1. *Inicialização.* Seja n_i e m_o o número de neurônios nas camadas de entrada e de saída, respectivamente.

$$t_{ij}(0) = 1$$

$$w_{ij}(0) = \frac{1}{1+n_i}$$

$$0 \le i \le n_i - 1 \qquad \text{e} \qquad 0 \le j \le m_o - 1$$

Fixe o valor do parâmetro de vigilância, ρ, tal que $0 < \rho < 1$.

2. *Aplique um novo padrão à rede*, $\boldsymbol{x}(t)$.

3. *Calcule a ativação*

$$\mu_j(t) = \sum_{i=0}^{n_i-1} w_{ij}(t) \cdot x_i(t), \qquad 0 \le j \le m_o - 1 \tag{4.1}$$

onde μ_j é a saída do neurônio j e x_i é o componente i do vetor de entrada $\boldsymbol{x}$, que pode ser tanto 0 quanto 1.

4. *Selecione o vencedor*, ou neurônio com maior ativação

$$\mu_{j^*}(t) = \max_j \left[\mu_j(t)\right], \qquad 0 \le j \le m_o - 1 \tag{4.2}$$

5. *Teste*:
$$\begin{cases} \text{Caso } \dfrac{\|\boldsymbol{T} \cdot \boldsymbol{x}\|}{\|\boldsymbol{T}\|} > \rho, \quad \text{vá para o passo 7.} \\ \text{Caso contrário vá para o passo 6.} \end{cases}$$

$$\text{onde } \|\boldsymbol{T}\cdot\boldsymbol{x}\| = \sum_{i=0}^{n_i-1} t_{ij^*}(t)\cdot x_i(t), \qquad \text{e} \qquad \|\boldsymbol{x}\| = \sum_{i=0}^{n_i-1} x_i(t).$$

6. *Desabilite o neurônio vencedor.* Faça a saída do neurônio vencedor, j^*, igual a zero. Vá para o passo 3.

7. *Adapte o neurônio vencedor*:

$$t_{ij^*}(t+1) = t_{ij^*}(t)\cdot x_i(t).$$
$$w_{ij^*}(t+1) = \frac{t_{ij^*}(t)\cdot x_i(t)}{0.5 + \sum_{i=0}^{n_i-1} t_{ij^*}(t)\cdot x_i(t)} \tag{4.3}$$

8. *Habilite neurônios desabilidatos, e vá para o passo 2.*

Para o caso em que a similaridade é menor que o parâmetro de vigilância, ρ, onde $0 < \rho < 1$, um sinal de *reset* é gerado, permitindo a criação de uma nova categoria associada a um neurônio de saída. Deve-se inicializar a ART com neurônios suficientes para acomodar o número esperado de agrupamentos a serem formados. Como descrito anteriormente, haverá ressonância quando um estímulo de entrada for suficientemente próximo a um protótipo já armazenado. Um dos problemas do ART é a escolha adequada do parâmetro de vigilância. Este valor tem ligação direta à dinâmica de geração de novas classes de padrões e a literatura cita vários casos onde uma mínima modificação neste valor ocasiona respostas completamente diferentes na estrutura de agrupamentos encontrada. Outro problema ocorre quando não há neurônios livres na camada de saída: um novo estímulo será desprezado. Isto poderia ser contornado, de forma relativamente simples, embora biologicamente não realista, com idéias derivadas das redes construtivas: poder-se-ia efetivar um mecanismo de geração de novos neurônios na camada de saída.

Além das desvantagens citadas no início da seção, os neurônios representantes dos agrupamentos encontrados neste modelo não apresentam relações explícitas de vizinhança. Não há formação de uma representação ordenada (informação topológica), como existe no SOM, podendo grupos de dados vizinhos no espaço p-dimensional serem representados por neurônios em posições quaisquer.

4.1.2 Neocognitron

Desenvolvido por K. Fukushima (1980), o "Neocognitron" , derivado do "Cognitron" (Fukushima, 1975), é uma RNA inspirada no trabalho pioneiro de Hubel e Wiesel (1962) na determinação da organização cortical do sistema visual. O Neocognitron é uma RNA hierárquica de múltiplas camadas e foi aplicado com relativo sucesso a problemas de reconhecimento de padrões visuais, como reconhecimento de caracteres manuscritos e impressos (dígitos). Fukushima et. al. (1983) mostraram que o Neocognitron é capaz de reconhecer caracteres de forma invariante à posição, apresentando assim, invariância translacional dos objetos no campo visual.

Seguindo Fukushima (1993), diversas restrições que controlam as redes neurais biológicas têm sido investigadas e introduzidas no projeto dessas RNAs. Estas restrições incluem conexões locais entre camadas numa rede hierárquica, campos receptivos não uniformes, conexões retroativas, conexões pulando camadas, atenção seletiva e conexões inibitórias. Há possibilidade de aprendizado tanto de forma supervisionada quanto não-supervisionada. O mecanismo básico de aprendizado competitivo do Neocognitron é derivado da regra de Hebb (1949).

No modelo mais simples (Fukushima 1980) existem três tipos de células: S, C, V (modelos mais recentes incluem outras como W). O estágio inicial (camada de entrada) consiste em uma matriz de neurônios receptores. Cada estágio seguinte tem uma camada de células S seguido de uma camada de células C, e assim por diante, i.e., alternam-se camadas de células S e C, como mostrado na figura 4.1. Entre as camadas existem pesos sinápticos excitatórios e inibitórios, que também podem ser fixos (como por exemplo os pesos que ligam os neurônios das camadas $S_n \rightarrow C_n$) ou variáveis (os pesos que ligam os neurônios das camadas $C_n \rightarrow S_n$). As conexões são efetuadas de acordo com o padrão *forward,* i.e., camadas U_{L-1} para camadas U_L. O Neocognitron usa a técnica de campos receptivos e combinação de atributos locais para gerar atributos mais globais em níveis mais elevados, i.e., camadas posteriores. As conexões de uma camada $C \rightarrow S$ são variáveis e são reforçadas durante o processo de treinamento. Atributos locais, como uma linha em uma orientação particular, ou uma intersecção de linhas são extraídas nas camadas iniciais. Atributos de natureza mais global são extraídas em estágios de ordem mais elevada, permitindo ao sistema reconhecer os padrões de entrada. As células do tipo S são neurônios que têm a função de extração de atributos, enquanto que as células C foram adicionadas (não havia células C no Cognitron), para compensar erros posicionais dos atributos, permitindo, desta forma, a invariância a translação dos padrões. A última camada de células

C é a camada de reconhecimento, ou classificação, e representa o resultado a uma dada classe de padrões.

Internamente, cada camada é dividida em sub-grupos de neurônios, de acordo com os atributos a que eles respondem. Cada sub-grupo é chamado de plano de células. No caso do treinamento supervisionado, deve-se fornecer a cada plano de células o tipo de atributo a ser aprendido. Para o caso não supervisionado, o algoritmo deve determinar quais atributos serão aprendidos pelos planos de células. As conexões convergindo a um plano de células são homogêneas, o que representa que cada neurônio em um plano recebe conexões de uma mesma distribuição espacial proveniente da camada anterior. Seguindo a teoria de que atributos mais complexos são processados em estágios de ordem mais elevada, o Neocognitron é projetado de forma que a densidade de células decresça em relação à ordem do estágio (podemos definir estágio como um par, em cascata, de camadas *S* e *C*). Outra característica é que células vizinhas recebem sinais similares.

Um diagrama do Neocognitron pode ser visto na figura 4.1. A arquitetura apresentada foi proposta por Fukushima (1988) para reconhecer 35 padrões diferentes: 26 letras e 9 dígitos (0 e O são tratados como o mesmo padrão). Note que invariância por rotação não foi considerada por nenhum modelo do Neocognitron.

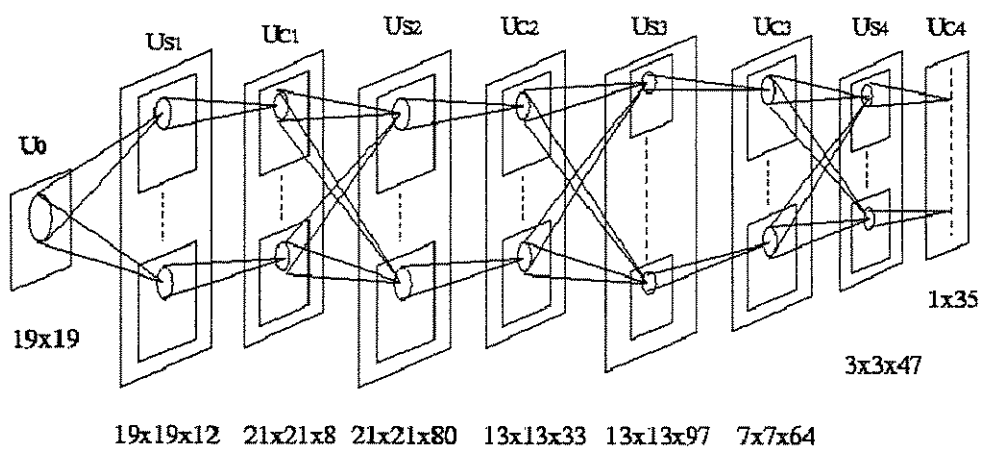


Figura 4.1: Um diagrama de uma RNA Neocognitron proposta para reconhecer 35 padrões alfa-numéricos. Adaptado de Fukushima (1988).

Entre as vantagens apresentadas por Fukushima está a habilidade de generalizar padrões, mesmo tendo treinado a rede com poucos padrões significativos. Outra vantagem é que não é necessário pré-processar a imagem, por exemplo para obter atributos ou normalizar os

padrões em relação a tamanho ou posição no campo visual. Entre as desvantagens estão o enorme número de conexões e neurônios (necessitando assim de uma grande capacidade de memória), e a necessidade de camadas e estruturas complexas, mesmo para resolver um problema simples como reconhecimento de 5 diferentes classes de dígitos. A tolerância à translação é alcançada às custas de replicação massiva de hardware.

Barnard e Casasent (1990) mostraram que o desempenho do Neocognitron não é intrinsecamente invariante a translações, e determinados parâmetros do modelo devem ser escolhidos apropriadamente para a obtenção de uma invariância translacional aproximada. Diversas variações têm sido aplicadas ao modelo básico. Fukushima incrementou o modelo de forma que fosse possível efetuar atenção seletiva sobre os padrões, quando um determinado estímulo visual apresenta dois ou mais padrões (Fukushima, 1991). O método utilizado foi reforçar o ganho de células que contribuem para um determinado padrão vencedor no último estágio, camada de reconhecimento, de forma que um mecanismo de memória associativa possa, através de conexões no sentido reverso do utilizado para reconhecimento, fazer aflorar, corretamente, um dos padrões contidos no estímulo de entrada. Uma vez este padrão é "restaurado" pelo mecanismo de memória associativa, o que lembra um pouco a rede ART, ele serve de entrada para a camada inicial (C_0). Uma vez o primeiro padrão reconhecido, os canais que foram reforçados (ganhos) são anulados e um outro padrão contido na cena aflora na camada de saída. O processo é refeito enquanto houver padrões na cena de entrada. Se no início do processo não há padrão classificado na camada de reconhecimento, um circuito de detecção de ausência de padrões é acionado, o que faz com que vários parâmetros da rede (limiares de células) sejam diminuídos. Outras implementações na literatura incluem Himes e Iñigo (1992), que apresentaram um sistema para reconhecimento automático de alvos militares utilizando o Neocognitron. Minnix et al. (1992) utilizaram a transformação de Walsh-Haddamard modificada para gerar representações invariantes de uma imagem e após uma Neocognitron modificada para reconhecer os padrões normalizados em posição. Ting & Chuang (1993) estenderam a habilidade do Neocognitron, através de um algoritmo adaptativo, para reconhecer imagens em níveis de cinza. Chao & Stoner (1993) implementaram o Neocognitron opticamente, conseguindo invariância a translações usando correlações ópticas multicanais de Fourier em cada camada de processamento. Processamento multi-camadas foi obtido interativamente retroagindo a saída do correlator de atributos para a entrada do modulador espacial de luz e atualizando os filtros de Fourier.

Apesar de interessante, o Neocognitron é bastante restrito à área de visão computacional, tendo sido modelado com propósito específico de reconhecimento de padrões tolerante a translação e pequenas deformações, o que é conseguido através de vários estágios compostos por camadas de células *C* e *S*.

4.1.3. Redes Neurais Gaussianas

Redes neurais Gaussianas (GNN) são similares a redes tipo RBF (Radial Basis Function), (Bishop, 1995), porém são treinadas por um algoritmo competitivo (Hamad et al., 1996; Firmin et al., 1997). Considera-se, neste modelo, a hipótese de que os agrupamentos sejam Gaussianos, e há grande semelhança com o método de identificação de misturas de funções de densidade de probabilidade, descrito no capítulo 2. A rede possui duas camadas, conectadas no padrão *feedforward*. Cada neurônio na primeira camada representa um componente da mistura, i.e., um agrupamento, enquanto que a camada de saída provê a estimação da densidade de probabilidades da mistura. O processo de treinamento é usado para calcular os vetores de média, as matrizes de covariâncias e as probabilidades de cada componente da mistura. Todos os neurônios da primeira camada possuem a mesma função de ativação, Gaussianas, e o número de neurônios é inicializado após testes com três critérios de informação: AIC, MDL e o critério do logaritmo da verossimilhança (LLC). O AIC (*Akaike information criterion*) é geralmente usado em análises de séries temporais para identificação de modelos (Akaike, 1974), enquanto que o MDL (*minimum description lenght*) é baseado em teoria de codificação (Rissanen, 1978). A camada de saída apresenta apenas um neurônio que representa a função densidade de probabilidade da mistura.

O princípio de treinamento da GNN é o uso de aprendizado competitivo com a distância de Mahalanobis. O ajuste dos parâmetros é feito apenas no neurônio vencedor para um dado padrão $\boldsymbol{x}(t)$, de acordo com

$$\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + \eta(t)\left[\boldsymbol{x}(t) - \boldsymbol{m}_i(t)\right] \tag{4.4}$$

$$\Sigma_i(t+1) = \Sigma_i(t) + \eta(t)\left\{\left[\boldsymbol{x}(t) - \boldsymbol{m}_i(t)\right]\cdot\left[\boldsymbol{x}(t) - \boldsymbol{m}_i(t)\right]^T - \Sigma_i(t)\right\} \tag{4.5}$$

onde i denota o índice do neurônio vencedor e $\eta(t)$ é a taxa de aprendizado, definida por

$$\eta(t) = \eta_0 \cdot \left(1 + t/t_0\right)^{-1} \tag{4.6}$$

e os parâmetros η_0 *e* t_0 devem ser fornecidos pelo usuário. O algoritmo descrito pelas equações 4.4-4.6 é conhecido como o método *search-then-converge* e suas propriedades foram descritas em Darken & Moody (1991). Os pesos conectando cada neurônio da primeira camada com o neurônio da segunda camada expressam a proporção de

importância de cada componente na mistura, sendo relacionado com as probabilidades *a priori* de cada componente. Eles são computados, ao final do processo de treinamento da primeira camada, como a razão do número de padrões classificados no componente i em relação ao número total de padrões usado para o treinamento (equação 4.7).

$$\pi_i = \frac{1}{N}\sum_{n=1}^{N} u_k(\boldsymbol{x}_n) \tag{4.7}$$

onde a função $u_k(\boldsymbol{x}_n)$ será 1 caso o padrão $\boldsymbol{x}_n$ pertença à classe k, e zero no caso contrário. N é o número de padrões no conjunto de dados X. Um esquema iterativo para determinar π_i seria

$$\pi_i(t+1) = \eta(t)\cdot u_k[\boldsymbol{x}(t)] + [1-\eta(t)\cdot\pi_i(t)] \ . \tag{4.8}$$

As equações 4.4-4.6 e 4.8 são usadas no processo de treinamento competitivo. O número de neurônios da primeira camada, K, é de extrema importância, sendo escolhido entre uma faixa de valores [K_{min}, K_{max}]. Como deve haver mais padrões do que parâmetros a serem estimados (ver seção 2.5), $N > (p+1)\cdot(p+2)\cdot K/2$, o que limita K_{max} em $(2N)/[(p+1)\cdot(p+2)]$. O valor de K_{min} é escolhido pelo usuário.

Em geral, os critérios de identificação de misturas são derivados do critério do logaritmo da verossimilhança e são brevemente descritos a seguir. Tais critérios têm sido usados no contexto estatístico dos modelos de identificação de misturas de Gaussianas, antes de usar o algoritmo EM (ver seção 2.5).

Seja $\hat{\Theta} = \left\{\hat{\theta} : \hat{\theta} = \left(\hat{\pi}_1, \hat{\mu}_1, \hat{\Sigma}_1, \hat{\pi}_2, \hat{\mu}_2, \hat{\Sigma}_2, \ldots, \hat{\pi}_k, \hat{\mu}_k, \hat{\Sigma}_k\right)\right\}$ uma estimação dos parâmetros $\Theta = \left\{\theta : \theta = \left(\pi_1, \mu_1, \Sigma_1, \pi_2, \mu_2, \Sigma_2, \ldots, \pi_k, \mu_k, \Sigma_k\right)\right\}$. Podemos escrever a função densidade de probabilidade $p(X)$ como uma função dependente dos parâmetros $\hat{\Theta}$, ou seja, $p(X, \hat{\Theta})$. A verossimilhança do conjunto de dados $\boldsymbol{X}$ é escrita como

$$L(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N) = \prod_{n=1}^{N} p\left(X, \hat{\Theta}\right). \tag{4.9}$$

e o logaritmo da verossimilhança é dado por

$$l\left(\hat{\Theta}\right) = \log \prod_{n=1}^{N} p\left(X, \hat{\Theta}\right) \equiv \sum_{n=1}^{N} \log p\left(X, \hat{\Theta}\right). \tag{4.10}$$

O LLC (*log-likelihood criterion*) é dado por

$$\mathrm{LLC} = -2 \cdot l\left(\hat{\Theta}\right) = -2 \cdot \sum_{n=1}^{N} \log\, p\left(X, \hat{\Theta}\right) \equiv -2 \cdot \sum_{n=1}^{N} \log \left[\sum_{k=1}^{K} \pi_k \cdot g_k\left(x_n; \hat{\mu}_k, \hat{\Sigma}_k\right) \right]. \tag{4.11}$$

Onde os parâmetros da mistura $\left(\hat{\mu}_k, \hat{\Sigma}_k\right)$, $k = 1, ..., K$, são determinados pelas equações 4.4-4.8. O inconveniente de usar este critério é que ele possui tendência a superestimar o número de neurônios K, que é o número de componentes da mistura. Akaike (1973) propôs usar uma penalização na equação (4.10), proporcional ao número de parâmetros independentes da mistura

$$\mathrm{AIC} = -2 \cdot l\left(\hat{\Theta}\right) + 2m \tag{4.12}$$

onde m é o número de parâmetros independentes a serem estimados da mistura. Bozdogan (1992, 1993) propôs que o termo de penalização fosse $3m$. À medida que o tamanho do conjunto de dados aumenta outros esquemas foram propostos para penalizar sobre-parametrizações. Por exemplo, Rissanen (1978, 1989) propôs um critério baseado em teoria de codificação que usa a informação estatística dos dados e dos parâmetros. O MDL é definido por

$$\mathrm{MDL} = -2 \cdot l\left(\hat{\Theta}\right) + m \cdot \log(N) \tag{4.13}$$

onde *log* (N) é o logaritmo natural do tamanho do conjunto de dados, N. Outros critérios de informação para identificação de misturas podem ser vistos em Bozdogan (1987, 1993).

Os resultados apresentados pelos autores para simulações numéricas foram satisfatórios. A GNN é praticamente o método convencional estatístico de identificação de misturas com uma roupagem neural. O problema continua sendo a determinação de um grande número de parâmetros, especialmente quando a matriz de covariâncias é não-diagonal. A escolha do número de componentes da mistura é crucial no método. Escolhendo-se este valor errado implica nos mesmos problemas dos métodos particionais: impor uma estrutura aos dados, no lugar de descobri-la.

4.1.4. Redes neurais competitivas hierárquicas (RNCH)

Alguns modelos hierárquicos foram propostos para redes neurais competitivas. Existem modelos estáticos e dinâmicos, estes últimos motivados pela possibilidade do próprio sistema ajustar automaticamente o número de neurônios para um determinado problema. Tais modelos dinâmicos, em geral, possuem heurísticas para crescimento e redução da rede, i.e., adição e eliminação de neurônios. O problema em geral é a seleção dos parâmetros externos que controlam a dinâmica da rede, que devem ser fornecidos no início do treinamento. Em geral o treinamento é feito *top-down*, possivelmente aproveitando os pesos estabelecidos na camada superior para inicializar a próxima camada, objetivando a redução do tempo de treinamento. Outra motivação para uso de RNCH é que a busca, por exemplo, de um neurônio vencedor, pode ser feita de forma mais eficiente, e além disto, durante o treinamento a adaptação fica restrita a ramos da árvore, o que implica em ganho de tempo e estabilidade em relação a redes competitivas convencionais. A seguir, faz-se uma breve descrição sobre alguns modelos de RNCH.

4.1.4.1 *SONT - Self-Organizing Neural Tree*

O modelo *Self-Organizing Neural Tree* (SONT) pré-define a arquitetura da rede antes do início do treinamento, podendo haver adição ou eliminação de neurônios após o treinamento (Li et al., 1993a). Três métodos de treinamento foram propostos. O primeiro efetua busca exaustiva sobre todos os neurônios, em cada nível da árvore, para encontrar o neurônio vencedor, *c*. Uma vez encontrado *c*, todos os membros de sua sub-árvore são adaptados. Os outros dois métodos usam uma busca ordenada em cada nível da árvore, pesquisando apenas na sub-árvore correspondente ao neurônio vencedor. Um método adapta apenas o neurônio vencedor enquanto o outro adapta toda a sub-árvore correspondente. A deficiência deste modelo é a rigidez da estrutura, escolhida *a priori*. Caso não tenha sido escolhida de forma compatível ao conjunto de dados, os resultados são insatisfatórios, principalmente em classificação automática de dados.

4.1.4.2 *DNTN - Dynamic Neural Tree Networks*

O modelo DNTN (*Dynamic Neural Tree Networks*) objetiva a flexibilização da estrutura da árvore de neurônios pelo uso de heurísticas para crescimento e poda da rede (Li et al., 1992; Racz & Klotz, 1991). Redes DNTNs são redes de uma camada, com padrão de conectividade *feed forward*, onde uma estrutura hierárquica é sobreposta aos neurônios da camada de saída. Os neurônios são criados dinamicamente e cada neurônio possui conexões à camada de entrada. Os neurônios da camada de saída são agrupados e o grupo de neurônios é organizado em uma estrutura tipo árvore. Os grupos de neurônios são ativados

um a cada vez, sendo que o grupo pertencente ao primeiro nível da árvore, a raiz, é o grupo que é ativo inicialmente. O neurônio vencedor em um determinado grupo é adaptado pelo método de aprendizado competitivo convencional (ver capítulo 3).

Existem dois conjuntos de parâmetros (tolerância e limiares) que influenciam a dinâmica de crescimento da rede. A tolerância define um raio de uma hiper-esfera de classificação, para cada neurônio e em cada nível da árvore. Quanto mais próximo da raiz, maior será o valor da tolerância de um neurônio. Este parâmetro tem função e problemas similares ao parâmetro de vigilância da rede ART. Caso a distância entre um padrão e um neurônio esteja abaixo deste valor, o neurônio é dado como vencedor.

O conjunto de parâmetros limiares são usados para determinar quando um grupo de neurônios deve ser criado. A rede de Li et al. (1992) armazena para cada neurônio o erro cumulativo que foi produzido após a classificação dos padrões. Caso este valor ultrapasse um limiar haverá uma expansão, i.e., geração de um sub-grupo a partir deste neurônio. O erro cumulativo pode ser definido como o somatório das distâncias de todos os padrões aos neurônios mais próximos, sendo o erro de quantização do conjunto de dados relacionado a um neurônio. A rede de Racz e Klotz (1991) compara a atividade de um neurônio com a atividade do neurônio pai. Haverá expansão caso a razão das atividades (quantas vezes o neurônio filho classifica padrões do neurônio pai) ultrapassa um limiar.

O algoritmo básico é descrito a seguir. Inicialmente a estrutura da árvore está vazia. Quando um padrão é inserido, um neurônio é criado no grupo raiz para classificar esta entrada. Os pesos sinápticos do neurônio raiz são fixados iguais ao primeiro padrão inserido na rede. Uma vez feita a seleção do conjunto de parâmetros para cada neurônio, tolerâncias e limiares, de forma *a priori*, a estrutura da árvore é gerada dinamicamente. À medida que apresentamos os padrões à rede, quatro situações podem ocorrer. Seja $d[\boldsymbol{x}(t), \boldsymbol{w}_i(t)]$ a distância entre o neurônio i e o padrão $\boldsymbol{x}(t)$. O neurônio vencedor em um grupo k, c^k, é o neurônio que possui menor distância ao padrão $\boldsymbol{x}(t)$. Seja a tolerância e o limiar de um neurônio i denotados, respectivamente, por ζ_i e o limiar υ_i.

1. Caso o neurônio vencedor do grupo atual, c^k, esteja dentro da tolerância (i.e., $d[\boldsymbol{x}(t), \boldsymbol{w}_{c^k}(t)] < \zeta_{c^k}$) e não possua neurônios filhos, este neurônio classifica o padrão e adapta seus pesos sinápticos usando a regra competitiva convencional, i.e., movendo-se na direção do padrão de acordo com uma taxa de aprendizado.
2. Caso o neurônio vencedor do grupo atual esteja dentro da tolerância e possua neurônios filhos, ele adapta seus pesos movendo-se na direção do padrão, e o grupo de neurônios filho torna-se o grupo atual.

3. As condições similares à situação 1, porém, o valor de limiar, $\upsilon_c{}^k$, é excedido. Haverá a criação de um novo sub-grupo.
4. Caso o neurônio vencedor não esteja dentro da tolerância, $d[\boldsymbol{x}(t), \boldsymbol{w}_c{}^k(t)] > \zeta_c{}^k$, um novo neurônio é criado no grupo atual, e seus pesos são inicializados de forma a coincidir com o padrão inserido, i.e., $\boldsymbol{w}_i{}^{new}(t) \leftarrow \boldsymbol{x}(t)$.

Note que, no caso 3, a árvore cresce em níveis, enquanto que no caso 4 há expansão no nível atual da árvore. Apesar deste modelo ser bastante interessante, e ser mais flexível do que a *SONT*, nenhuma heurística foi fornecida pelos autores para seleção de parâmetros que possam guiar a dinâmica da rede a construir uma estrutura representativa do conjunto de dados. A escolha dos conjuntos de parâmetros tem grande influência no resultado final, e podem produzir árvores bastante não balanceadas, com número excessivo de níveis, mesmo quando os dados não apresentam tal estrutura. Estas redes são também bastante sensíveis a outliers, o que ocasiona a criação de neurônios para cada padrão que dista mais que o valor pré-estabelecido de tolerância.

4.1.4.3 *CENT - Competitive Evolutionary Neural Tree*

A rede CENT (*Competitive Evolutionary Neural Tree*) foi desenvolvida de modo similar ao modelo DNTN, porém, neste caso, parâmetros como os limiares e as tolerâncias são calculados a partir de regras heurísticas, eliminando a necessidade de escolhas a *priori* (Adams et al., 1999). O que irá fazer um neurônio se dividir será a sua atividade relativa dentro do grupo. Pode haver expansões em níveis da rede ou lateralmente, no mesmo grupo. Cada operação de crescimento da rede é avaliada, e caso não tenha desempenho satisfatório, ela é desfeita, i.e., o neurônio é eliminado.

O treinamento inicia com apenas um neurônio (raiz) que tem seus pesos inicializados como a média do conjunto de dados. O erro cumulativo de cada unidade é definido de forma similar ao DNTN: soma de todas as distâncias do neurônio aos padrões que foram classificados por ele, o que dá um indicativo da variância do subconjunto de dados associado ao neurônio. A tolerância inicial do neurônio raiz é feita de forma que dois terços do conjunto de dados distem menos que este valor.

O que dita a dinâmica da rede é a atividade do neurônio. Quando o neurônio classifica um padrão sua atividade é incrementada em uma unidade. Caso contrário, decresce-se da atividade uma fração de forma a que caso o neurônio não classifique nenhum padrão em pelo menos um terço de época, sua atividade será zero. Esta forma de decaimento da

atividade é similar ao apresentado em Li et al. (1992). A seleção inicial para fazer crescer a rede não implica diretamente no crescimento lateral do grupo ou de novos grupos. Porém, um cuidado foi tomado em relação a *outliers*: só há crescimento em neurônios que classificam um subgrupo de dados, o que não ocorre na DNTN. As condições para crescimento do neurônio incluem a proporção de classificação de padrões em relação aos outros neurônios do mesmo grupo, que deve ser elevada, e que não tenha sido escolhido anteriormente para uma expansão que posteriormente foi desfeita. As condições de crescimento para um neurônio são referidas como maturidade (μ), potencial de crescimento (ρ) e atividade (δ).

Um neurônio é considerado maduro quando existe há pelo menos M épocas de treinamento. Os autores consideraram o valor 1/15 do total do número de épocas, limitando inferiormente em 75. Inicialmente todo neurônio possui um potencial de crescimento não nulo, porém caso haja um determinado número de tentativas (η) não bem sucedidas de crescimento ρ é anulado, o que inibe quaisquer futuras tentativas de crescimento a partir daquele neurônio. O valor de η sugerido pelos autores foi 4. A atividade do neurônio é similar ao limiar no modelo DNTN. Um neurônio i é considerado suficientemente ativo caso

$$\delta(i) > \frac{\delta(i')}{|N_i|+\varepsilon} \tag{4.14}$$

onde i' é o neurônio pai de i, e $\delta(i')$ é sua atividade correspondente. O valor $|N_i|$ corresponde à cardinalidade do grupo de neurônios ao qual i faz parte, enquanto que ε é uma constante, para a qual os autores sugerem o valor 2. Caso a atividade de um neurônio exceda a atividade média dos neurônios do grupo ele é selecionado para crescimento. Apenas neurônios folhas, i.e., que estejam no nível mais inferior em um sub-ramo da árvore são considerados para crescimento. A tolerância do neurônio (ζ), definida como o raio da hiperesfera na qual os padrões podem ser classificados ou não dentro da área de influência do neurônio, é que determina se haverá crescimento lateral (no mesmo grupo) ou para baixo (adição de um novo nível). Caso vários padrões estejam fora do raio de tolerância, novos neurônios são adicionados no mesmo nível, i.e., no mesmo grupo, de forma que possam representar melhor o subgrupo de dados.

Os critérios para aceitação / rejeição dos neurônios adicionados à rede são a soma dos erros quadráticos (SEQ) e a atividade dos novos neurônios, que deve ser acima que um valor estabelecido. O desempenho da expansão é analisada no próximo terço de época, onde comparam-se a soma dos erros quadráticos do neurônio pai (i') e do filho (i). Caso SEQ(i) seja menor que SEQ(i') e a atividade do neurônio i, $\delta(i)$, seja não nula, a operação de

crescimento é mantida. Cada neurônio filho herda o vetor de pesos bastante similar ao vetor de pesos do neurônio pai: $w_i = w_{i'} + \lambda$, onde λ é um vetor com a mesma dimensão de w de ruído Gaussiano com média zero e variância 0.25 vezes a tolerância do neurônio pai, i.e., $\zeta_{i'} / 4$. A tolerância do neurônio filho é definida como

$$\zeta(i) = \zeta(i') \cdot \frac{2 \cdot |\psi(i)| + |\vartheta(i)|}{3 \cdot |\psi(i)|} \tag{4.15}$$

onde $|\psi(i)|$ é a cardinalidade (o número de padrões) do subconjunto classificados dentro da tolerância do neurônio i. Matematicamente, teríamos que $\forall\ x \in \psi(i) \Leftrightarrow \| x - w_i \| < \zeta_i$. $\vartheta(i)$ denota o subconjunto de padrões onde i é o neurônio mais próximo, porém a distância destes padrões ao neurônio excede a tolerância.

Alguns resultados da CENT são apresentados e discutidos no capítulo 6, onde haverá comparações com um modelo hierárquico proposto nesta tese. Apesar de, no início do artigo, os autores afirmarem que o modelo possui dinâmica livre de parâmetros, havendo necessidade apenas do conjunto de dados, o que houve em relação ao modelo DNTN foi uma troca de seleção *a priori* de limiares e tolerâncias por uma série de outros parâmetros (M, η, ε, δ, ζ, ...) e fórmulas heurísticas para fazer o crescimento da rede. Os próprios autores (Adams et al., 1999, p. 546) afirmam a necessidade de mais estudos sobre as implicações da escolha destes parâmetros.

4.1.5. Outros modelos

O modelo SPAN (*space partition network*) adiciona neurônios em uma grade bidimensional (Lee e Peterson, 1990). Um neurônio é dividido em dois, caso sua contribuição para o erro total da rede exceda um limiar pré-estabelecido. O novo neurônio é posicionado na vizinhança do neurônio original, caso exista posição livre. Caso contrário, a grade é estendida para acomodá-lo. O grande problema desta rede é a sensibilidade à ordem de apresentação dos dados, que pode fazer com que a rede cresça de forma bastante diferente quando fazemos duas ou mais simulações.

Outro modelo inspirado em sistemas biológicos e com aplicação em sistemas de visão computacional é a rede *Dynamic Link Architecture* (DLA) de von der Malsburg (Wiskott & von der Malsburg,1996). DLA foi proposta para mapeamento de padrões similar ao SOM, porém mais complexa, e tem sido aplicada em vários problemas, principalmente em visão

computacional, como reconhecimento de objetos invariante a transformações geométricas e reconhecimento de rostos humanos (Buhmann *et al.*, 1990).

4.2 Modelos derivados do SOM

Alguns modelos de redes neurais foram propostos para CA tendo como base o algoritmo e funcionalidades do SOM. Em geral, as modificações objetivam a obtenção de diversos mapas separados, ou regiões do mapa que representem classes, seja pela divisão de mapas durante o treinamento, ou pelo uso de vários mapas desde o início do treinamento.

4.2.1 *Incremental Grid Growing*

O método *Incremental Grid Growing* (IGG) foi desenvolvido por J. Blackmore e R. Miikkulainen (1993, 1995), com o objetivo de solucionar dois problemas do SOM convencional: (1) a necessidade de definir o tamanho do mapa *a priori*; e (2) detectar a borda dos agrupamentos. Geralmente, IGG inicia com um mapa de tamanho 2×2 neurônios e durante o processo de treinamento novos neurônios são adicionados em posições do mapa onde há grande concentração de padrões, caso haja um erro cumulativo elevado nesta posição. Por exemplo, olhando para a figura 3.32, o neurônio que está representando o agrupamento no lado direito da figura seria um candidato a dividir seus padrões com um novo neurônio que seria alocado para diminuir o erro, representado na figura 3.32 como linhas conectando os padrões ao neurônio vencedor. Desta forma, o objetivo do IGG é crescer em tamanho com objetivo de representar melhor os dados. Conexões entre neurônios vizinhos são adicionadas ou eliminadas em função da distância entre os pesos sinápticos, definida no espaço de entrada. A conseqüência disto é que podem ocorrer subdivisões do mapa, e várias sub-redes consistindo de conjuntos de neurônios bastante similares, representando os diferentes agrupamentos dos dados.

O algoritmo de aprendizado é baseado no algoritmo convencional do SOM, seção 3.4, com apenas alguns passos a mais para permitir o processo de crescimento.

1. Um padrão de entrada $\boldsymbol{x}_k = (\xi_1, \xi_2, \ldots, \xi_p)$, $\boldsymbol{x}_k \in \Re^p$, é selecionado aleatoriamente de todo o conjunto de padrões.

2. Uma função de ativação é usada para calcular o estado de cada neurônio $\boldsymbol{m}_i$ em relação ao padrão $\boldsymbol{x}_k$.

3. Seleciona-se o neurônio vencedor, c, por exemplo, com a equação 3.5.

4. Os pesos sinápticos do neurônio vencedor, c, como também os pesos dos neurônios que estão dentro da vizinhança de c são atualizados (por exemplo, equação 3.6).

5. Repita os passos 1-4 até completar a fase de organização da rede.

6. Para todos os neurônios em posições de fronteira da rede, analise o erro cumulativo $\boldsymbol{E}_i$.

7. Adicione novos neurônios na vizinhança dos neurônios na fronteira da rede que apresentem erros cumulativos mais elevados, e inicialize seus pesos sinápticos.

8. Adicione ou elimine conexões entre neurônios de acordo com a distância de seus pesos sinápticos.

Neurônios em posições de fronteira (passo 6, 7) são definidos como neurônios que não estão complemente conectados a outros neurônios, i.e., pelo menos deve haver uma conexão lateral aberta. O algoritmo IGG inicia-se efetivamente após o término dos passos 1-4. Os passos 6-8 não são processados ciclicamente, mas em fases organizacionais. O treinamento é dividido em diversas fases, cada uma consistindo de um certo número de iterações, cada uma possuindo um conjunto específico de parâmetros para função de vizinhança, taxa de aprendizado e valores limiares para adição ou eliminação de conexões entre neurônios.

Durante o processo de treinamento, o erro cumulativo $\boldsymbol{E}_i(t+1)$ para todos os neurônios i em posições de fronteira é calculado pela soma dos erros cumulativos dos neurônios $\boldsymbol{E}_i(t)$ e pela distância quadrática entre o padrão $\boldsymbol{x}_k$ e o vetor de pesos do neurônio vencedor.

$$\boldsymbol{E}_i(t+1) = \boldsymbol{E}_i(t) + \sum_{k=1}^{p} \left(\xi_k - \boldsymbol{m}_{ik}(t)\right)^2 \tag{4.16}$$

Geralmente neurônios que representam um grande número de padrões diferentes apresentarão valores elevados para o erro cumulativo, $\boldsymbol{E}_i(t)$. Novos neurônios são adicionados em todas as posições de vizinhança abertas do neurônio que apresente o maior valor de erro, o qual denominaremos de *err*. Para poder gerar uma estrutura em 2D, apenas

os neurônios em posições de fronteira, ou no perímetro da rede, são consideradas no processo de crescimento. Após a adição de novos neurônios, seus pesos são inicializados através das equações 4.17 e 4.18. Caso já existam neurônios na vizinhança do novo neurônio adicionado, seus pesos são inicializados pela média de todos os pesos sinápticos dos neurônios circunvizinhos (N_{novo}), equação 4.17, onde $|N_{novo}|$ expressa a cardinalidade do conjunto de neurônios circunvizinhos ao neurônio sendo adicionado.

$$\boldsymbol{m}_{novo,k} = \frac{1}{|N_{novo}|} \sum_{i \in N_{new}} \boldsymbol{m}_{ik} \tag{4.17}$$

Caso não existam neurônios circunvizinhos ao novo neurônio, os pesos sinápticos são inicializados com

$$\boldsymbol{m}_{novo,k} = \boldsymbol{m}_{err,k} \cdot (|N_{err}| + 1) - \sum_{i \in N_{err}} \boldsymbol{m}_{ik} \tag{4.18}$$

onde N_{err} representa o conjunto de todos os neurônios circunvizinhos ao neurônio *err*. Desta forma, os pesos do neurônio adicionado é composto por parcelas dos pesos do neurônio *err* e de seus vizinhos.

Adicionando novos neurônios em regiões com grande número de padrões permite uma melhor representação do espaço de entrada. Não há estratégia para exclusão de neurônios, apenas para eliminação de conexões laterais entre estes, quando a distância entre dois neurônios vizinhos ultrapassa um limiar t_d. Neurônios podem também ser conectados, caso a distância de seus pesos estiver abaixo de um outro limiar t_c. Ao final do processo de treinamento, o efeito de adição de neurônios e conexões laterais, como também o efeito de eliminação destas últimas, podem gerar mapas completamente separados. Um dos grandes problemas neste método é suprir, antes do início do treinamento, o conjunto de parâmetros, incluindo os parâmetros básicos do SOM e diversos outros como limiares t_c, t_d, etc., sendo seu desempenho bastante susceptível às várias escolhas. Outro ponto é que a ordem de apresentação dos dados pode influir no resultado final.

4.2.2 *Growing Cell Structures*

Seguindo o mesmo princípio do IGG, o algoritmo *Growing Cell Structures* (GCS) baseia-se no princípio de inicializar um mapa pequeno, de dimensão 2, e adicionar novos neurônios em regiões que apresentem grande concentração de padrões, de modo a representar adequadamente o espaço de entrada dos dados. Entretanto, neste modelo, podem-se adicionar como também eliminar neurônios durante o processo de treinamento. O GCS foi desenvolvido inicialmente por B. Fritzke (1993, 1994, 1995, 1996), e nesta seção descreveremos brevemente as idéias concernentes ao algoritmo. De forma similar ao IGG, após o treinamento o GCS geralmente divide o mapa em várias redes distintas e completamente separadas, cada uma representando uma classe específica de padrões de entrada.

Uma diferença em relação ao IGG é o padrão de conectividade entre os neurônios, que no GCS é triangular. Inicialmente o mapa consiste de apenas 3 neurônios formando um triângulo. Durante o processo de treinamento, novos neurônios são adicionados em posições que recebem grande número de padrões. Diferentemente do IGG, estes novos neurônios não são adicionados no perímetro da rede e sim dentro do mapa. O ajuste de conexões entre os neurônios é feito de forma a sempre preservar o padrão triangular de conectividade na rede. Após adição de neurônios, algumas iterações são efetuadas para permitir a reordenação da rede. Neurônios que após o processo de treinamento recebam poucos ou nenhum padrão (baixo grau de atividade, ver seção 3.6) são eliminados juntamente com suas conexões, o que pode ocasionar um rompimento da rede em diversas sub-redes menores. Estas sub-redes podem continuar crescendo e dividindo-se independentemente das outras.

De forma similar ao IGG, o processo de treinamento ocorre em fases organizacionais diferentes. Após a ordenação, via o algoritmo convencional do SOM, a distribuição dos padrões nos neurônios é verificada, sendo que pode-se adicionar ou eliminar neurônios. Cada neurônio possui um contador, que é incrementado quando ele vence a competição por um padrão, e decrementado, por uma fração, no caso contrário. O processo básico de treinamento é descrito abaixo.

1. Um padrão de entrada $\boldsymbol{x}_k = (\xi_1, \xi_2, \ldots, \xi_p)$, $\boldsymbol{x}_k \in \Re^p$, é selecionado aleatoriamente de todo o conjunto de padrões.

2. Uma função de ativação é usada para calcular o estado de cada neurônio $\boldsymbol{m}_i$ em relação ao padrão $\boldsymbol{x}_k$.

3. Seleciona-se o neurônio vencedor, c, por exemplo, com a equação 3.5.

4. Os pesos sinápticos do neurônio vencedor, c, como também os pesos dos neurônios que estão dentro da vizinhança de c são atualizados de acordo com a estrutura triangular.

5. Modifique os contadores de padrões, τ_i.

6. Repita os passos 1-5 até completar a fase de organização da rede.

7. Adicione novos neurônios em posições com elevada densidade de padrões, inicializando os pesos sinápticos e os contadores de padrões. Elimine neurônios que não estejam recebendo padrões, i.e., $\tau_i = 0$.

8. Inicie uma nova fase organizacional.

Durante o treinamento, cada vez que um neurônio i é vencedor, seu contador τ_i é incrementado em uma unidade. O contador dos outros neurônios é decrescido de um fator α. O objetivo disto é enfatizar neurônios que estejam ganhando mais nas épocas mais recentes. A equação 4.19 ilustra a operação sobre os contadores.

$$\begin{cases} \tau_i(t+1) = \tau_i(t)+1, & \text{caso neurônio } i \text{ seja o vencedor, } c. \\ \tau_i(t+1) = \tau_i(t)\cdot(1-\alpha), & \text{caso contrário, i.e., } \forall i, i \neq c. \end{cases} \tag{4.19}$$

Após a finalização de cada fase organizacional, que dura um certo número de épocas de treinamento, o neurônio q que possui o valor máximo do contador é selecionado. Um novo neurônio é adicionado entre q e um outro neurônio r (ver equação 4.20) que esteja na vizinhança imediata de q, N_q, e que seja o neurônio mais distante, considerando o espaço de pesos. N_q compreende os neurônios que estão conectados diretamente ao neurônio q.

$$\forall p \in N_q, p \neq r, r \in N_q: \qquad \left\| \boldsymbol{m}_q - \boldsymbol{m}_r \right\| \geq \left\| \boldsymbol{m}_q - \boldsymbol{m}_p \right\| \tag{4.20}$$

Os pesos do novo neurônio adicionado são inicializados com a média dos pesos dos neurônios vizinhos, q e r: $\boldsymbol{m}_{novo} = \frac{1}{2}(\boldsymbol{m}_q - \boldsymbol{m}_r)$.

O contador de padrões do novo neurônio, τ_{novo}, é inicializado de forma a representar uma estimativa da distribuição dos dados junto aos neurônios. Isto é feito considerando o

tamanho da região de decisão de cada neurônio, sendo a região de Voronoï V_p do neurônio p dada por

$$\forall i \in N_p : \quad V_p = \frac{1}{|N_p|} \cdot \sum_i \|\boldsymbol{m}_p - \boldsymbol{m}_i\| \tag{4.21}$$

onde $|N_p|$ é a cardinalidade do conjunto de neurônios circunvizinhos ao neurônio p.

O objetivo é modificar os contadores τ de todos os neurônios circunvizinhos ao neurônio adicionado de tal modo que, depois das modificações, os contadores de todas as unidades possuam valores próximos aos que eles teriam possivelmente tido se o novo neurônio estivesse presente durante a última fase organizacional do treinamento. Isto é obtido modificando os contadores de todos os neurônios circunvizinhos ao novo neurônio:

$$\forall i \in N_{novo} : \quad \tau_i(t+1) = \tau_i(t) + \frac{V_i(t+1) - V_i(t)}{V_i(t)} \cdot \tau_i(t) \tag{4.22}$$

E o contador de padrões do novo neurônio pode ser expresso como o somatório das mudanças dos contadores dos neurônios circunvizinhos:

$$\tau_{novo}(t+1) = -\sum_{i \in N_{novo}} \Delta\tau_i(t) \equiv \frac{V_i(t+1) - V_i(t)}{V_i(t)} \cdot \tau_i(t) \tag{4.23}$$

A eliminação de neurônios ocorre quando o contador de um dado neurônio está abaixo de um valor limiar ν. O cuidado que deve ser tomado na eliminação de neurônios é em relação à estrutura triangular das vizinhanças dos neurônios da rede, que deve ser mantida.

As deficiências do modelo GCS incluem principalmente a escolha dos vários parâmetros a serem selecionados antes do início do treinamento. Kohonen et al. (1996c) descreveram que GCS foram muito mais susceptíveis a variações nas escolhas dos parâmetros iniciais que o SOM convencional. Mesmo usando uma grande variedade de valores de parâmetros o GCS foi incapaz de produzir resultados significantes. Assim, o GCS pode, evidentemente, produzir bons resultados, mas quando temos um conjunto de dados novo podemos ter dificuldades em usá-lo, principalmente para a detecção de características desconhecidas presentes neste conjunto de dados.

4.2.3 Múltiplos mapas bidimensionais (*Multiple 2-D SOM*)

W. Wan e D. Fraser (1993, 1994a-c) apresentaram um modelo baseado em múltiplos mapas auto-organizáveis bidimensionais, o qual foi chamado de M2dSOM. A idéia básica é usar um algoritmo de treinamento ligeiramente modificado de forma que cada um dos mapas represente um agrupamento específico dos padrões. Isto é obtido tratando cada mapa de forma independente uma vez que o vencedor foi selecionado.

Idealmente, cada agrupamento de dados deveria ser representado por um mapa diferente. Porém, isto requer um certo grau de conhecimento *a priori* sobre o número de agrupamentos esperado no conjunto de dados para selecionar um número apropriado de mapas e assim obter resultados satisfatórios. O algoritmo de aprendizado é apresentado a seguir.

1. Um padrão de entrada $\boldsymbol{x}_k = (\xi_1, \xi_2, \ldots, \xi_p)$, $\boldsymbol{x}_k \in \Re^p$, é selecionado aleatoriamente de todo o conjunto de padrões.

2. Uma função de ativação é usada para calcular o estado de cada neurônio $\boldsymbol{m}_i$ em relação ao padrão $\boldsymbol{x}_k$.

3. Seleciona-se o neurônio vencedor, c, e o mapa vencedor Ψ_i, como o SOM que contém o neurônio c ($c \in \Psi_i$).

4. Os pesos sinápticos do neurônio vencedor, c, como também os pesos dos neurônios que estão dentro da vizinhança de c são atualizados (só o mapa Ψ_i) de acordo com a regra de aprendizado *local*.

5. Crie o conjunto $N_{r,c}(t)$ dos próximos r vencedores e adapte os pesos dos neurônios que não tiveram modificação durante o passo de aprendizado *local*.

Uma vez tendo inserido um padrão, todos os mapas concorrem, de forma independente, em níveis de ativação. O neurônio que possuir maior ativação, e o seu respectivo mapa, são denominados vencedores daquele padrão naquele instante de tempo. O critério de encontrar o vencedor é o mesmo do SOM convencional (capítulo 3). O mapa que possui o neurônio vencedor é treinado como se fosse o único SOM em questão. No conjunto $N_{r,c}(t)$ podem existir unidades que estejam em mapas vizinhos, e a vizinhança em questão é no espaço de pesos e não usando o espaço do grid do SOM. Estes neurônios vizinhos que não foram atualizados pela regra de aprendizado local são atualizados de forma similar aos neurônios

do mapa vencedor Ψ_i, apenas usando-se uma taxa de aprendizado $\alpha_x(t)$ inferior ao usado no mapa $\Psi_i, \alpha_x(t) < \alpha(t)$.

Uma vez concluído o processo de treinamento, os k mapas representam os k agrupamentos esperados. Porém, não há relação a ser descoberta entre os mapas vizinhos, por exemplo. Estes são independentes. Caso aumentemos o valor de k, pode-se obter um resultado que seja a subdivisão de um ou mais conjuntos de dados que estavam sendo representados por um ou mais mapas, porém, nenhuma informação sobre super-agrupamentos pode ser obtida neste método. Novamente, a escolha de parâmetros depende de conhecimento *a priori* sobre a distruibuição dos agrupamentos. A estrutura global é perdida quando aumenta-se a granularidade do M2dSOM, i.e., um agrupamento fica disperso em dois ou mais mapas independentes, o que é bastante indesejável.

4.2.4 SOM Hierárquico (HSOM)

O uso de vários mapas de Kohonen com estrutura hierárquica foi proposto por R. Miikkulainen (1990, 1991, 1993a). Diferentemente do M2dSOM, os mapas não são independentes entre si, mas possuem uma estrutura hierárquica, rígida, com a forma de uma pirâmide. A idéia é que agrupamentos diferentes sejam mapeados em mapas diferentes no mais baixo nível de hierarquia, e ao mesmo tempo reduza o esforço computacional de treinamento da rede.

Iniciando na raiz da árvore, ou no topo da pirâmide, sub-mapas são criados para cada neurônio em um determinado nível. Por exemplo, podemos ter um HSOM que no primeiro nível tenha um mapa com tamanho 2×2. Cada neurônio neste mapa pode estar conectado a um sub-mapa de tamanho 2×2 no segundo nível, que por sua vez possuem neurônios conectados a outros 16 sub-mapas no terceiro nível, de tamanho $P \times Q$.

Um dos problemas do método é que o tamanho da pirâmide, i.e. o número de níveis como também o tamanho dos mapas a cada nível, deve ser decidido *a priori*, o que significa que não existe nenhum mecanismo de crescimento dinâmico de mapas novos. Além disto, todos os mapas em um determinado nível da pirâmide possuem o mesmo tamanho, mesmo que estejam representando quantidades bastante diferentes de padrões. O treinamento do HSOM é executado seqüencialmente, i.e., do nível mais elevado da pirâmide até o mais baixo. Outra característica é que geralmente escolhe-se tamanhos pequenos para mapas em níveis iniciais, como citado no parágrafo anterior, ex. 2×2. O algoritmo básico de treinamento é descrito a seguir:

1. Fixe o nível de treinamento atual em $n = 0$.

2. Fixe o mapa atual C como o mapa de nível mais elevado.

3. Um padrão de entrada $\boldsymbol{x}_k = (\xi_1, \xi_2, \ldots, \xi_p)$, $\boldsymbol{x}_k \in \Re^p$, é selecionado aleatoriamente de todo o conjunto de padrões.

4. Uma função de ativação é usada para calcular o estado de cada neurônio $\boldsymbol{m}_i$ em relação ao padrão $\boldsymbol{x}_k$ no mapa atual C.

5. Seleciona-se o neurônio vencedor, c, no mapa atual C.

6. (*a*) se o nível de $C < n$, i.e. o mapa atual é ainda mais elevado em hierarquia do que o nível atual a ser treinado, selecione o mapa subordinado do neurônio vencedor, c, como mapa atual C e continue no passo 4.

   (*b*) caso contrário, i.e. se o nível $C = n$, já estamos no nível a ser treinado: Os pesos sinápticos do neurônio vencedor, c, como também os pesos dos neurônios que estão dentro da vizinhança de c são atualizados (algoritmo convencional do SOM).

7. Repetem-se os passos 2-6 até que o processo de treinamento convirja no nível n.

8. Se não foi alcançado ainda o nível mais alto da hierarquia continue no passo 2 com o próximo nível, i.e., faça $n = n + 1$.

O processo de adaptação é aplicado subseqüentemente a cada nível do mapa, a partir do mapa com nível mais elevado. Pode-se usar combinações diferentes de parâmetros de treinamento em cada nível. A regra de treinamento é idêntica ao algoritmo básico do SOM, com a exceção de que, neste caso, o treinamento é efetuado para mapas diferentes em cada nível. Após a convergência de todos os mapas em um determinado nível, o próximo nível da pirâmide pode ser treinado. Neurônios que recebam um único padrão de entrada não necessitam ter seus mapas subsequentes treinados. Da mesma forma quando a variância dos objetos mapeados em um neurônio estiver abaixo de um valor limiar, indicando que estes são bastante similares, cessa-se a necessidade de treinar mapas subsequentes.

Este modelo permite a criação de sub-mapas que possuam relações hierárquicas com mapas em níveis superiores, porém uma das maiores vantagens é em relação ao ganho de tempo

no treinamento em relação a um SOM convencional, se considerássemos um único mapa com tamanho equivalente ao tamanho da base da pirâmide. Outras formas de ganhar tempo no processo de treinamento do SOM foram descritas na seção 3.5.3, onde podemos iniciar um mapa pequeno, treiná-lo e ir inserindo neurônios de forma a manter sempre a ordenação obtida com o mapa de tamanho inferior. Na seção 3.5.3, também descreve-se um método que permite diminuir o tempo de treinamento pela simples diminuição da área de busca do neurônio vencedor à medida que o treinamento avança. Estas duas abordagens, que podem ser aplicadas em conjunto, permitem um ganho de tempo bastante razoável na fase de treinamento (Costa & Netto, 1999c).

De forma similar ao M2dSOM, IGG, e GCS, todos os parâmetros são escolhidos *a priori*, como o número máximo de níveis, n, os tamanhos dos mapas em cada nível, valores limiares, além dos parâmetros de cada mapa, como ocorre no algoritmo convencional, incluindo taxa de aprendizado, tamanho da vizinhança inicial e formas de decaímento, etc. Escolhas erradas de tamanhos de mapas podem levar a resultados inadequados. Apesar de ser uma evolução em relação ao M2dSOM, bordas de agrupamentos no HSOM ainda precisam ser desenhadas manualmente.

4.2.5 TS-SOM - *Tree structured SOM*

O modelo TS-SOM (Koikkalainen, 1994) representa uma implementação semelhante ao modelo HSOM, com a estrutura estática e piramidal, onde cada neurônio possui quatro outros neurônios filhos (ver figura 4.2). A rede é treinada seqüencialmente por níveis, do topo até a base da pirâmide. Pode-se usar pesos de um nível para inicializar os pesos do nível inferior, com o objetivo de reduzir o tempo de treinamento. Este modelo foi projetado com o objetivo de aumentar a velocidade do treinamento, principalmente no que diz respeito à busca pelo neurônio vencedor. A busca por árvore é mais eficiente do que a busca seqüencial, principalmente quando a árvore está balanceada. Para N neurônios têm-se uma complexidade computacional de $O(log\ N)$ contra $O(N)$ do método convencional. A adaptação dos pesos pode ser feita localmente na sub-árvore correspondente. O algoritmo de treinamento é bastante parecido com o HSOM (seção 4.2.4), o que difere é que no HSOM cada nó da árvore é uma rede de Kohonen, de tamanho fixo, enquanto que neste caso cada nó da árvore é um neurônio. No TS-SOM, todo um nível é treinado como sendo uma rede única, enquanto que no HSOM o treinamento ocorre apenas na sub-rede que teve o neurônio pai vencedor para o padrão inserido.

Os resultados em simulações com dados provenientes de distribuições uniformes foram semelhantes ao resultado do SOM convencional. As deficiências deste modelo para

classificação automática são similares às apresentadas nas redes anteriores, como a HSOM, principalmente pela rigidez da estrutura que é imposta já no início do treinamento.

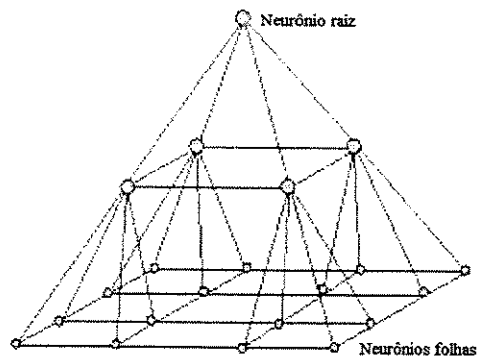


Figura 4.2: A estrutura piramidal do TS-SOM

4.2.6 SCONN - *Self-Creating and Organizing Neural Networks*

O modelo SCONN (*Self-Creating and Organizing Neural Networks*) foi proposto por Choi e Park (1994) com o objetivo de criação de quantizadores vetoriais que, de forma similar ao GCS e IGG, adicione novos neurônio para representar melhor o espaço dos dados. Porém, este modelo inicia com apenas um neurônio, e via um processo de decaimento de um nível de ativação dos neurônios existentes, geram-se novos neurônios, porém, mantendo sempre o objetivo de não possuir neurônios mortos[1] na configuração final obtida. O algoritmo básico do SCONN é descrito a seguir.

1. Inicialize os pesos sinápticos (w_i).

2. Apresente um padrão de entrada $x_k = (\xi_1, \xi_2, \ldots, \xi_p)$, $x_k \in \Re^p$, selecionado aleatoriamente de todo o conjunto de padrões.

3. Calcule a distância $\| w_i - x_k \|$ para todos os neurônios i.

[1] Como ressaltado no capítulo 3, neurônios mortos correspondem a neurônios que no final do treinamento permaneceram em regiões com densidade de pontos nulas, não possuindo, desta forma, nenhum objeto no domínio de suas células de Voronoï.

4. Selecione o neurônio vencedor, c, como o que possui a distância mínima ao padrão x_k dentre todos os neurônios, i.e., $\| x - w_c \| = \min_i \{ \| x - w_i \| \}$, i = 1, 2, ..., $N_n(t)$, onde $N_n(t)$ é o número de neurônios no instante t.

5. Decida se o neurônio vencedor, c, está ativo. Caso positivo, vá para o passo 6. Caso contrário vá para o passo 7.

6. Adapte os pesos do neurônio vencedor (ou pode-se usar um esquema como ocorre no SOM, onde adaptamos também neurônios na vizinhança do vencedor). Decresça o nível de ativação de todos os neurônios. Volte para o passo 2.

7. Adicione um novo neurônio a partir do neurônio vencedor inativo, onde neste caso inatividade corresponde ao nível de ativação do neurônio estar abaixo de um valor limiar estabelecido. Decresça o nível de ativação de todos os neurônios. Volte para o passo 2.

O nível de ativação, $\theta(t)$, é um dos parâmetros mais importantes neste modelo. No início todos os neurônios possuem valores elevados para serem ativados por qualquer padrão inserido. Decrescendo y com o tempo ocorrerá que neurônios com grande freqüência de vencimentos irão gerar neurônio 'filhos'. O passo 5 determina se o neurônio vencedor está ou não ativo. Caso $\| w_c - x_k \| < \theta(t)$, o neurônio c é considerado ativo, sendo inativo caso contrário. Os pesos do neurônio vencedor, e ativo, são adaptados da mesma forma do algoritmo de treinamento do SOM (equação 3.5), porém, neste caso, pode-se adaptar apenas os pesos do vencedor, c.

O processo de criação de neurônios (passo 7) é feito da seguinte forma:

$$w_j(t+1) = w_i(t) + \lambda(t) \cdot \left(x(t) - w_i(t) \right)$$

onde $0 < \lambda(t) < 1$, é um parâmetro que controla a semelhança entre o neurônio pai (w_i) e o neurônio filho, (w_j). À medida que $\lambda(t)$ aproxima-se de 1, decresce a semelhança de $w_j(t)$ em relação a $w_i(t)$.

Existem três formas básicas de parar o treinamento: (*i*) estabelecendo um número máximo de iterações; (*ii*) estabelecendo um número máximo de neurônios na rede; e (*iii*) estabelecendo um valor mínimo para o nível de ativação, $\theta(t)$. Os autores comentam que esta última estratégia é a mais adequada, pois o SCONN irá procurar automaticamente o número de neurônios adequado para o problema em questão.

Apesar de ser um modelo bastante interessante, os próprios autores consideram que há necessidade de conhecer o conjunto de dados para que se possa escolher adequadamente os vários parâmetros, como $\theta(t)$, $\lambda(t)$, etc., seus valores iniciais e suas formas de decaimento com o tempo. Mesmo objetivando não possuir neurônios mortos, os autores descrevem que em alguns casos isto ocorreu. Uma diferença fundamental em relação ao SOM, e aos modelos IGG e GCS é que os neurônios adicionados possuem relação de vizinhança apenas com o neurônio 'pai'. Não há a preocupação de construção de um mapa bidimensional, acomodando neurônios e conexões laterais de forma a manter a topologia da rede, como ocorre no GCS e IGG. Ainda, os autores constataram que o SCONN era em geral 1.5 vezes mais rápido para treinamento do que o SOM, em condições similares. A principal razão disto é que no início existem poucos neurônios, o que simplifica os cálculos de distâncias e busca pelo vencedor (passos 3 e 4). Uma pequena variação no algoritmo básico foi efetuada para permitir quantização não uniforme do espaço de dados, o que não ocorre no SCONN. O SCONN2 basicamente difere do SCONN no passo 6: decresce-se o nível de ativação $\theta(t)$ dos neurônios ativos e aumenta-se o $\theta(t)$ dos neurônios inativos.

4.3 Modelos baseados em interpretação do mapa

4.3.1 Coordenadas Adaptativas

Merkl (1997) e Merkl & Rauber (1997a,b) propuseram que as coordenadas[2] dos neurônios, por exemplo no caso bidimensional, $\langle ax_i, ay_i \rangle$, da mesma forma que os pesos sinápticos, que sofrem variações durante o aprendizado, pudessem ser influenciadas ao longo do treinamento, deslocando-se de forma a capturar, no espaço de saída ao final do treinamento, concentrações de neurônios, e desta forma poder, visualmente, detectar agrupamentos.

A técnica ´Coordenadas Adaptativas´ é, desta forma, uma proposta de visualização de possíveis agrupamentos, com o intuito de facilitar a análise da existência de agrupamentos de neurônios.

Inicialmente, no início do treinamento, as coordenadas de cada neurônio i são idênticas às coordenadas definidas na inicialização convencional de uma rede SOM. Durante cada passo do aprendizado (ver capítulo 3), a distância entre cada padrão apresentado $\boldsymbol{x}$ e o conjunto de pesos de cada neurônio é calculada e armazenada em uma tabela, $Dist(t)$. Após a adaptação

[2] Índices ou endereços dos neurônios no espaço de saída da rede.

dos pesos (eq. 3-6), uma nova distância é calculada, $Dist(t+1)$. A mudança relativa, $\Delta Dist_i(t+1)$, na distância para cada neurônio i da rede é calculada através da expressão

$$\Delta Dist_i(t+1) = \frac{Dist_i(t) - Dist_i(t+1)}{Dist_i(t)} \tag{4.24}$$

O movimento dos vários pesos sinápticos no espaço de entrada devido ao processo de adaptação é efetuado de forma similar no espaço de saída 'virtual' da rede. A adaptação na coordenada ax (o cálculo para as outras coordenadas é similar) é descrita por

$$ax_i(t+1) = ax_i(t) + \Delta Dist_i(t+1) \cdot [ax_c(t) - ax_i(t)] \tag{4.25}$$

onde $\langle ax_c, ay_c \rangle$ corresponde às coordenadas do neurônio vencedor, c.

Ao final do treinamento, pode-se visualizar cada neurônio i usando as coordenadas $\langle ax_i, ay_i \rangle$.

Coordenadas adaptativas além de não interferir no algoritmo convencional do SOM, provê, pelo uso da própria dinâmica do aprendizado, um método de rápida visualização da existência de agrupamentos. Porém, os autores alertam que o método deve ser apenas usado em fases avançadas do treinamento, i.e., após o final da fase inicial de ordenação, por exemplo, quando o raio de vizinhança ao redor do neurônio vencedor for a metade do valor inicial (Merkl, 1997). O uso do procedimento já no início do treinamento pode resultar em informações não úteis, devido ao fato de que no início do treinamento há grandes deslocamentos de pesos devido a fatores como o raio de vizinhança e a taxa de aprendizado serem elevadas. Tal técnica é efetiva apenas para mapas de dimensionalidade 1 ou 2.

4.3.2 Conexões entre agrupamentos

Merkl & Rauber (1997a,b) descrevem a técnica de pós-processamento do SOM '*Conexões entre Agrupamentos*' (Cluster Connections) como forma de obter informações contidas nos pesos sinápticos de um SOM treinado.

O método consiste em definir um conjunto de valores limiares que possam explicar os relacionamentos entre neurônios adjacentes. Os autores definiram três classes de relacionamentos entre neurônios: (1) altamente similares; (2) com similaridade intermediária; e (3) não similares. Desta forma, três valores limiares são utilizados.

A representação da classe de relacionamento entre neurônios adjacentes é feita através de linhas conectando os neurônios utilizando uma escala de cinza monotônica. Por exemplo, neurônios bastante similares são conectados por linhas pretas, enquanto que neurônios com similaridade intermediária são conectados por linhas cinzas. Neurônios não similares são conectados por linhas brancas, que ficam, desta forma desprezíveis em uma imagem com fundo branco.

A idéia do uso de limiares, a qual será também empregada no SL-SOM (capítulo 5) usando a *U-matrix*, objetiva identificar neurônios semelhantes (no espaço de pesos) e criar uma forma de representar esta informação para visualização. Um problema adicional, não comentado pelos autores, é a escolha dos limiares. Da mesma forma que o caso das coordenadas adaptativas, tal método é útil apenas quando o mapa possui dimensão 1 ou 2, e a decisão sobre a partição do mapa é novamente deixada como uma tarefa manual, i.e., requer intervenção do usuário.

4.3.3 Outras abordagens

Inspirado na forma que a *U-matrix* capta graficamente as relações de distâncias entre neurônios adjacentes (ver capítulo 5), Murtagh (1995) propôs o uso de um método hierárquico aglomerativo em um SOM treinado. O autor descreve o uso do método centróide e dois critérios de aglomeração de agrupamentos, a distância mínima ou a minimização do aumento da variância, que pode ser visto como o método Ward (ver seção 2.4.1.3). A escolha de tal método foi defendida pelo autor pela sua preferência por métodos que geram um representante único para o agrupamento após cada união. Aplicado a uma base de dados astronômica (IRAS PSC) contendo cerca de 250 mil registros (cada um descrito por quatro variáveis), o autor descreve a convergência do SOM já em seis épocas de treinamento. Apesar de usar um método hierárquico, não foi utilizada nenhuma estratégia de parada como descritas em Milligan & Copper (1985), tendo sido escolhido manualmente o número de agrupamentos, utilizando informações conhecidas sobre a base de dados.

Kaski *et al.* (1998b) propuseram dois métodos de interpretação de mapas treinados, também inspirados na informação contida na *U-matrix*. O primeiro refere-se a fatores locais que são aproximações (locais) por hiperplanos lineares dos neurônios a partir do uso de componentes principais. Os fatores são ajustados no espaço e representam os neurônios dentro de um certo raio centrado em cada neurônio. O segundo método, mais simples, é a

simples correlação entre a informação contida na *U-matrix* e cada componente (ou dimensão do espaço de pesos) do SOM. Desta forma, pode-se detectar componentes que possuam maior influência nos vales e bordas da *U-matrix* e representar, visualmente, apenas tais componentes que apresentem maior correlação. O primeiro método é bastante similar à visualização convencional dos componentes do SOM, e o segundo usa explicitamente a informação dos componentes, detectando a contribuição das variáveis na estrutura dos agrupamentos. Mesmo sendo métodos que provêem uma informação a mais que a *U-matrix*, isolada, todo o processo de agrupamentos ainda é deixado de forma manual, i.e., há necessidade de interferência manual por parte de um usuário que visualmente pode conduzir a separação dos grupos de neurônios. Os autores recomendam o uso de tais técnicas em áreas como análise exploratória de dados, por exemplo, em problemas onde buscam-se detectar características novas, e inesperadas, a partir dos dados, e em *mineração de dados*.

Rauber & Merkl (1999) apresentam um método com inspiração bastante relacionada com Kaski *et al.* (1998b), com o objetivo de detectar quais atributos são mais relevantes, em um mapa treinado, para atribuição de um conjunto de dados em um determinado agrupamento, i.e., um neurônio. Os autores descrevem que os métodos apresentados em Kaski *et al.* (1998b) requerem forte interação manual para examinar cada dimensão separadamente. O algoritmo *LabelSOM* determina a contribuição de cada elemento do vetor em relação à distância Euclidiana entre padrões e neurônios. Para cada neurônio, e seu conjunto de padrões associados, o erro de quantização para cada atributo individual serve como guia para suas relevâncias como rótulos das classes. Os mapas são apresentados na forma convencional (por exemplo, retangular), porém cada neurônio recebe rótulos (que são os nomes das variáveis ou atributos) em ordem decrescente da importância daquela dimensão para tal neurônio. Tal representação é usada em geral em conjunto com a técnica convencional de rotulagem do mapa treinado usando informação da classe dos itens mais freqüentes em cada neurônio. Porém, informações como número de agrupamentos, e suas bordas, não são discutidas no artigo, e todo o processo deve ser feito manualmente, possivelmente guiada com a ajuda visual da *U-matrix*.

4.4 Sumário

Foram descritos alguns modelos de redes neurais competitivas[3], incluindo alguns modelos variantes (ou derivados) do SOM. Alguns destes modelos, idéias e deficiências relacionadas servem de base e motivação para os métodos apresentados nos capítulos

posteriores. Em alguns casos haverá comparações de resultados dos métodos apresentados neste capítulo com os métodos propostos nesta tese.

A escolha de parâmetros nas redes competitivas é muitas vezes compreendido como um processo com algum componente supervisionado, e que em muitos casos pode levar a soluções adequadas, principalmente quando o usuário conhece as características da base de dados. Métodos construtivos propõem o uso de critérios como erro de quantização (ou variância dos padrões atribuídos a um neurônio) para expandir, ou mesmo diminuir uma rede, podendo inclusive gerar redes desconectadas, como é o caso da IGG. Modelos hierárquicos também foram descritos, porém em geral a estrutura é rígida, sendo determinada no início do treinamento, não refletindo, desta forma, as características dos dados.

Finalmente, métodos de interpretação de mapas treinados, apresentados na seção 4.3, geralmente inspiram-se na informação da *U-matrix*, porém requerem bastante interferência do usuário, sendo apenas ferramentas que auxiliam na decisão de como segmentar manualmente um SOM. O capítulo seguinte ilustra propostas de efetuar tal procedimento de uma forma automática, sugerindo o uso de técnicas de morfologia matemática (*watersheds*) e a detecção automática de marcadores, para segmentar a imagem representada na *U-matrix*.

[3] Atualmente existem milhares de modelos competitivos, sendo impossível uma abordagem exaustiva em apenas um capítulo.

Capítulo 5

Segmentação e rotulação automática dos mapas de Kohonen: O algoritmo SL-SOM

Este capítulo apresenta um método de particionamento e rotulação automática do SOM baseado na segmentação da imagem do gradiente (*U-matrix*) de todos os componentes de um mapa treinado. Descreve-se o algoritmo da *U-matrix* e comenta-se brevemente suas propriedades. A segmentação eficiente da *U-matrix* é proposta por meio do algoritmo *watershed*, que fornece uma abordagem simples para encontrar bons marcadores. Vários exemplos ilustram o processo de geração dos conjuntos de protótipos que representarão as geometrias dos agrupamentos de dados no espaço p-dimensional. Diferenças em relação a outros métodos, como misturas de funções densidades de probabilidades, são discutidas.

5.1 Visualização do mapas de Kohonen - A *U-matrix*

O SOM é usado para mapear um espaço de entrada p-dimensional, contendo n padrões, para uma grade de neurônios uni- ou bidimensional, com os objetivos de quantizar o espaço de entrada e representar da melhor forma possível a topologia original em um espaço de menor dimensão. Uma vez treinado, podemos rotular neurônios caso tenhamos, além da informação dos padrões, as suas respectivas classes. Padrões são apresentados ao mapa e o neurônio vencedor será o mais similar, ou o mais próximo, de acordo com o critério de similaridade escolhido. Diferentemente de PCA, onde o resultado da projeção é contínua, a projeção dos dados no SOM é discreta. O resultado de cada apresentação é um índice (i, j) denotando o neurônio vencedor.

Para efetuar análise de agrupamentos, este mapeamento topologicamente ordenado geralmente não é suficiente, pois a informação de distâncias entre os neurônios é perdida. A saída de um SOM para um dado padrão inserido é geralmente o índice do neurônio vencedor c, no caso bidimensional um par de valores (i, j), e o nível de ativação diretamente relacionado à quantização (no nosso caso a distância do padrão ao neurônio c, computado no espaço de pesos).

A visualização das relações entre os neurônios para um problema em que o vetor de entrada possui uma dimensão maior que 2 ou 3 torna-se bastante difícil. A representação tradicional do *grid*, Diagrama de Voronoï e/ou superfície de influências, deve ser substituída por métodos alternativos.

Um método de visualização de um SOM treinado, denominado a matriz de distâncias unificadas, ou *U-matrix*, foi desenvolvido por A. Ultsch (1993a,b) com o objetivo de permitir a detecção visual das relações topológicas dos neurônios. A idéia básica é usar a mesma métrica que foi utilizada durante o treinamento para calcular distâncias entre pesos sinápticos de neurônios adjacentes. O resultado é uma imagem $f(x, y)$, na qual as coordenadas de cada pixel (x, y) são derivadas das coordenadas dos neurônios no grid do mapa, ex. $(1,1)$, $(1, 2)$ ... $(X, Y) \rightarrow (1,1)$, $(1, 2)$ ... $(2*X-1, 2*Y-1)$, e a intensidade de cada *pixel* na imagem $f(x, y)$ corresponde a uma distância calculada. Pode-se pensar uma imagem como uma função tridimensional em que o valor do pixel na coordenada (x, y) é representado por um ponto na coordenada z. Neste caso, teremos uma superfície em 3D cuja topografia revela a configuração dos neurônios obtida pelo treinamento. Vales, neste relevo topográfico, correspondem a regiões de neurônios que são similares, enquanto que montanhas, i.e., valores relativamente elevados na *U-matrix*, refletem a dissimilaridade entre neurônios vizinhos e podem ser associadas a regiões ou neurônios em fronteiras de agrupamentos. Regiões de vales serão candidatas para representar agrupamentos de neurônios. Pelo fato de geralmente a *U-matrix* ser uma imagem relativamente complexa, principalmente em problemas de análise de dados reais, geralmente seu uso é restrito a visualização, sendo uma ferramenta de auxílio na separação manual dos agrupamentos de um SOM.

Apesar da descrição a seguir ser para um mapa cujos neurônios possuem vizinhança retangular, a lógica pode ser estendida, por exemplo para vizinhança hexagonal. Considere um mapa retangular de tamanho $X \times Y$. Seja $[b_{x,y}]$ a matriz de neurônios e $[wi_{x,y}]$ a matriz de pesos. Para cada neurônio em b existem três distâncias d_x, d_y e d_{xy}, na *U-matrix*, a seus vizinhos (ver figuras 5.1 e 5.2).

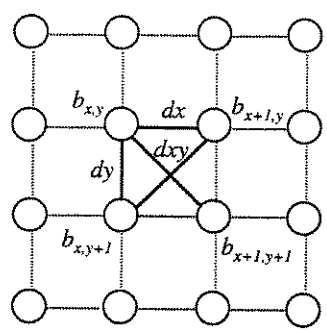


Figura 5.1: As três distâncias da U-matrix

Considerando distâncias Euclidianas, no caso da topologia da grade de neurônios ser retangular, as distâncias d_x, d_y e d_{xy} podem ser definidas como

$$dx(x,y) = \left\| b_{x,y} - b_{x+1,y} \right\| = \sqrt{\sum_i \left(w_{i_{x,y}} - w_{i_{x+1,y}} \right)^2} \tag{5.1}$$

$$dy(x,y) = \left\| b_{x,y} - b_{x,y+1} \right\| = \sqrt{\sum_i \left(w_{i_{x,y}} - w_{i_{x,y+1}} \right)^2} \tag{5.2}$$

$$\begin{aligned} dxy(x,y) &= \frac{1}{2}\left(\frac{\left\| b_{x,y} - b_{x+1,y+1} \right\|}{\sqrt{2}} + \frac{\left\| b_{x,y+1} - b_{x+1,y} \right\|}{\sqrt{2}} \right) \\ &= \frac{1}{2\sqrt{2}}\left[\sqrt{\sum_i \left(w_{i_{x,y}} - w_{i_{x+1,y+1}} \right)^2} + \sqrt{\sum_i \left(w_{i_{x,y+1}} - w_{i_{x+1,y}} \right)^2} \right] \end{aligned} \tag{5.3}$$

Estas distâncias, calculadas no espaço dos pesos, são plotadas em uma matriz U de tamanho (2X-1) × (2Y-1). A *U-matrix* combina as três distâncias, ou gradientes, considerando todos os componentes do SOM. Para cada neurônio de b, as distâncias para os vizinhos (se estas existirem) tornam-se dx, dy e dxy, e a U-Matrix é preenchida de acordo com a tabela,

TABELA 5.1 - ESQUEMA PARA PREENCHIMENTO DOS ELEMENTOS DA U-MATRIX

i	j	(i,j)	$U_{i,j}$
I	P	$(2x+1, 2y)$	$dx(x,y)$
P	I	$(2x, 2y+1)$	$dy(x,y)$
I	I	$(2x+1, 2y+1)$	$dxy(x,y)$
P	P	$(2x, 2y)$	$du(x,y)$

onde as abreviações *'I'* e *'P'* referem-se ao índice ou posição do neurônio, sendo ímpar e par, respectivamente. A equação 5.4 apresenta os elementos da *U-matrix*, preenchida de acordo com a tabela 5.1.

$$\begin{bmatrix} du(0,0) & dx(0,0) & du(1,0) & \dots & du(X\text{-}1,0) \\ dy(0,0) & dxy(0,0) & dy(1,0) & \dots & dy(X\text{-}1,0) \\ du(0,1) & dx(0,1) & du(1,1) & \dots & du(X\text{-}1,1) \\ dy(0,1) & dxy(0,1) & dy(1,1) & \dots & dy(X\text{-}1,1) \\ \dots & \dots & \dots & \dots & \dots \\ du(0,Y\text{-}1) & du(0,Y\text{-}1) & du(1,Y\text{-}1) & \dots & du(X\text{-}1,Y\text{-}1) \end{bmatrix} \tag{5.4}$$

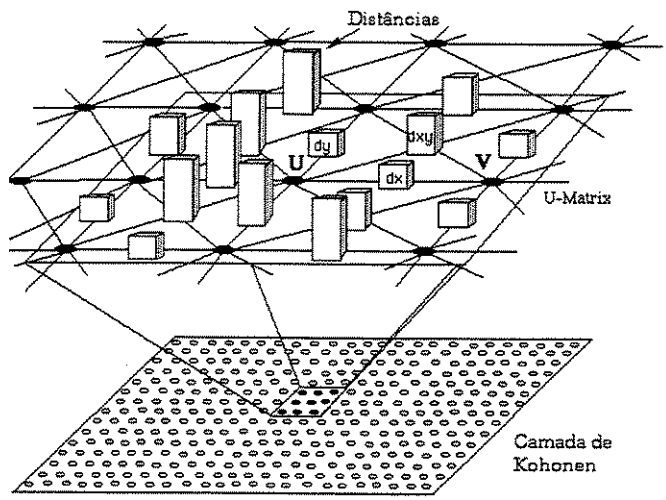


Figura 5.2: representação 3D da U-matrix (adaptado de Ultsch (1993a) com permissão)

Existem pelo menos duas possibilidades para o cálculo de $du(x, y)$: pode-se usar o valor médio ou a mediana dos elementos circunvizinhos. Seja $C = (c_1, c_2, \dots c_k)$ os valores dos elementos circunvizinhos de $U_{2x,2y}$ aparecendo na forma de um vetor ordenado com cardinalidade k ($k = |C|$). No nosso caso de topologia retangular, $k = 8$. No caso de $du(x, y)$ ser o valor mediano, temos

$$du(x,y) = \begin{cases} c\lceil (k+1)/2 \rceil & \text{se } k \text{ for } \textit{ímpar} \\ \dfrac{c(k/2) + c\lceil (k+1)/2 \rceil}{2} & \text{se } k \text{ for } \textit{par} \end{cases}$$

onde $c(k)$, $k = 1, 2, \dots K$, $K \leq 8$, denotam os elementos circunvizinhos ordenados de forma crescente de magnitude.

No caso de $du(x, y)$ ser o valor médio de C, temos

$$du(x, y) = \tilde{c} = \frac{1}{k}\sum_{i=1}^{k} c_i \,.$$

Para o SOM de tamanho *10×10* apresentado na figura 3.20, treinado com 1000 iterações do algoritmo em lote, e inicializado de forma linear, a *U-matrix* correspondente é mostrada na figura 5.3, na forma 3D, e na figura 5.4, na forma de uma imagem bidimensional. Note que as distâncias foram escalonadas linearmente, de forma que a distância máxima seja *1* e a mínima *0*. Olhando a figura 5.4 percebe-se a existência de três grandes regiões, uma à esquerda, e duas à direita, sendo uma na parte superior e a outra na parte inferior da figura. As bordas entre estas regiões, bastante detectáveis ao olho humano, devem-se principalmente à natureza relativamente simples do problema, i.e., os agrupamentos gerados artificialmente possuem uma separação tal que foi possível ao SOM obter uma configuração que concentrasse neurônios nas regiões dos agrupamentos, sobrando poucos neurônios em regiões de baixa densidade. Problemas em que os agrupamentos estão menos separados causam uma degradação das bordas da *U-matrix*, o que pode tornar sua segmentação extremamente difícil.

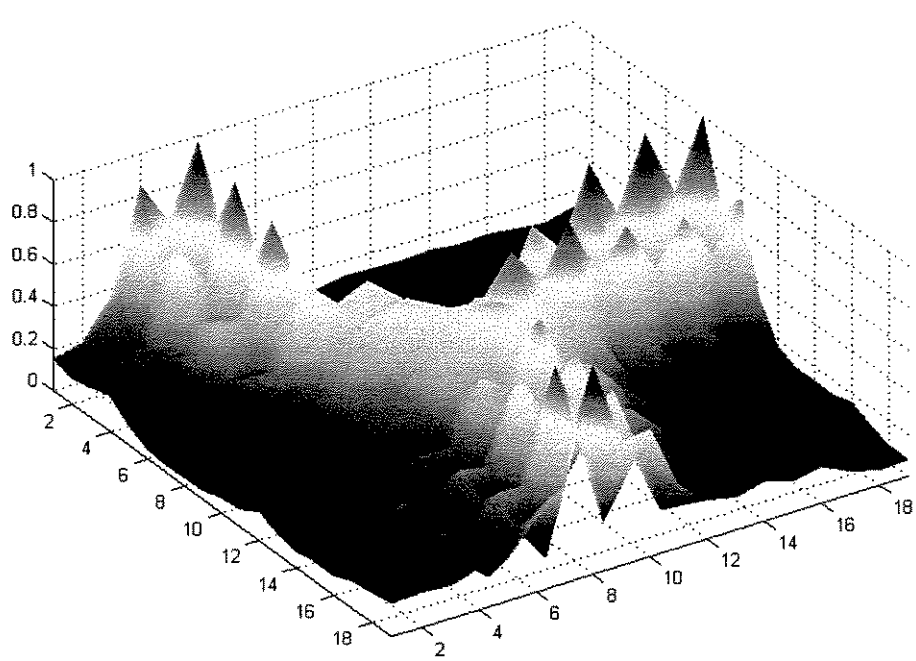


Figura 5.3: U-matrix na forma 3D do SOM 10x10 apresentado na figura 3.20

Mesmo neste problema relativamente simples, pode-se notar que em certas regiões as bordas são mais fortes do que em outras. Isto deve-se aos neurônios de ligação da grade que podem inclusive estarem inativos. Olhando para a figura 3.22 pode-se perceber neurônios (marcados com o símbolo +) que não estão representando nenhum objeto do conjunto de dados, entretanto foram usados nos cálculos da *U-matrix*, o que sem dúvida contribuiu para uma pequena degradação das bordas em regiões centrais, por exemplo (veja a figura 5.3).

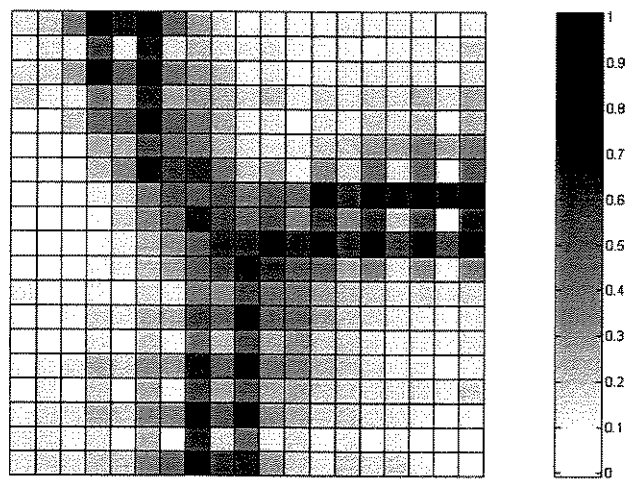


Figura 5.4: U-matrix, na forma de imagem bidimensional, do SOM 10x10 apresentado na figura 3.20.

Um modo de atenuar o efeito de neurônios inativos é tentar isolá-los do processo de cálculo da *U-matrix*, porém, a eliminação destes neurônios causariam buracos na rede. Uma solução para este problema é mostrada na seção 5.6.

5.2 Segmentação de Imagens

Esta seção tem o objetivo de abordar brevemente o assunto de segmentação de imagens. Atualmente existem diversas técnicas, geralmente aplicadas, cada uma, a um subconjunto de aplicações, não havendo um método universal para o processo de segmentação, principalmente por haver diversos tipos de imagens, obtidas por uma grande variedade de sensores, e suas várias formas de representação. Para uma abordagem mais completa, podem ser consultados livros e artigos específicos, como por exemplo, Gonzales e Woods (1992), Russ (1995), e Parker (1997).

Um dos grandes problemas em processamento e análise de imagens é a segmentação de seus componentes. Diversas tarefas de visão computacional são realizadas após a

segmentação, e assim a minimização de erros é crucial para um bom sistema de inspeção visual automática. Basicamente, segmentar uma imagem consiste em subdividir a imagem em suas partes ou objetos constituintes. Imagens tratadas por computador são discretizadas tanto em dimensões quanto em intensidade. A representação convencional de uma imagem é considerá-la uma função $f(x, y)$, onde amplitude f nas coordenadas x e y é proporcional à intensidade ou brilho, denominado nível de cinza quando a imagem é monocromática[1]. Geralmente, convenciona-se que x e y pertencem ao conjunto de números inteiros, ($\forall\ x, y \in \mathbb{Z}$), e consideraremos os limites para as coordenadas x e y como $[0, x_{max}]$ e $[0, y_{max}]$. Em relação a f, também discreta, contém uma faixa de valores entre um intervalo f_{min} e f_{max}, dependente da aplicação. O número de níveis diferentes de intensidades, n_i depende da discretização obtida na imagem. No caso mais comum, e que é usado nesta tese, $[f_{min}, f_{max}]$ [2] $= [0, 255]$ e n_i é 1, ou seja, há 256 níveis de cinza na imagem f. Portanto neste trabalho as imagens consideradas são monocromáticas e geradas artificialmente, veja por exemplo as figuras 5.3 e 5.4, onde busca-se segmentar as regiões de vale que estão associadas aos agrupamentos de neurônios no espaço p-dimensional.

Cada ponto da imagem, ou *pixel*, $f(x_i, y_i)$, onde $x_i \in [0, x_{max}]$ e $y_i \in [0, y_{max}]$, possui um conjunto de pixels vizinhos. A conectividade entre pixels é um conceito extremamente importante principalmente quando trata-se de segmentação de objetos e componentes de regiões em uma imagem. Os casos mais comuns consideram que os pixels estejam conectados aos quatro vizinhos (nas direções horizontal e vertical), ou aos oito vizinhos (os quatro anteriores mais os quatro pixels nas direções diagonais). Seja p um pixel nas coordenadas (x_i, y_i), o conjunto de pixels conectados a p no padrão de conectividade 4 são os pixels nas coordenadas (x_i+1, y_i), (x_i-1, y_i), (x_i, y_i+1) e (x_i, y_i-1). Este conjunto de pixels é denominado os 4 vizinhos de p, denotado por $N_4(p)$. No caso do padrão de conectividade 8, além dos pixels nas coordenadas citadas para $N_4(p)$, incluem-se os pixels nas coordenadas (x_i+1, y_i+1), (x_i+1, y_i-1), (x_i-1, y_i+1) e (x_i-1, y_i-1), formando o conjunto dos 8 vizinhos do pixel p, ou $N_8(p)$, (Gonzales e Woods, 1992). Alguns destes elementos podem estar fora dos limites físicos da imagem, $[0, x_{max}] \times [0, y_{max}]$, ocorrendo quando as coordenadas do pixel p estiverem em alguma posição da borda da imagem. No nosso caso em particular, os conjuntos dos vizinhos dos pixels situados na borda da imagem terão menos elementos do que os elementos que estão, por exemplo, em posições centrais da imagem. Por exemplo, caso p seja $(0, 0)$, o conjunto $N_4(p)$ possuirá apenas os pixels (x_i+1, y_i) e (x_i, y_i+1).

Obviamente, não há restrições à escolha do padrão de conectividade entre pixels, e pode-se escolher, de acordo com a aplicação, vários tipos de conectividade. O espaço dos pontos

[1] Imagens podem possuir múltiplas bandas, como é o caso de imagens coloridas, por exemplo no padrão RGB, ou ainda como exemplo imagens de satélites.

[2] Também denominada escala de níveis de cinza.

(x_i, y_i), $\mathbf{E} \equiv [0, x_{max}] \times [0, y_{max}]$, é dito 4-adjacente ou 8-adjacente caso seus pixels tenham relação de vizinhança 4 ou 8, respectivamente. Um pixel p é adjacente a um outro pixel q caso eles estejam conectados, i.e., $q \in N_m(p)$, onde m dita o padrão de conectividade. De forma similar, duas regiões R_1 e R_2 são adjacentes caso exista algum pixel de R_1 adjacente a um pixel de R_2. Um caminho de um pixel p com coordenadas (x_a, y_b) a outro pixel com coordenadas (x_c, y_d) é uma seqüência de pixels distintos $\{(x_a, y_b), \ldots, (x_c, y_d)\}$ que sejam adjacentes, na ordem da seqüência, e o comprimento do caminho é definido como o número de pixels da seqüência menos 1. Uma maneira de definir a distância entre dois pixels p e q, denotada por $d(p, q)$, é o comprimento do menor caminho de p a q.

Seguindo Gonzales e Woods (1992), se p e q são pixels de uma região, ou subconjunto R da imagem, então p está conectado a q em R caso exista um caminho de p a q consistindo inteiramente de pixels em R. Para qualquer pixel p em R, o conjunto de pixels em R que está conectado a p é denominado um componente conectado de R. Assim, dois pixels de um componente conectado estão conectados um ao outro, e que componentes conectados distintos são disjuntos.

Em geral, as técnicas de segmentação de imagens podem ser categorizadas em duas grandes classes de métodos que são técnicas derivadas de técnicas de extração de contornos e de técnicas de crescimento de regiões[3]. No primeiro caso, são usadas as descontinuidades, i.e., alterações bruscas nos níveis de cinza da imagem, e tenta-se traçar contornos entre regiões observando o máximo gradiente, ou uma função deste, ao longo de um caminho. As formas mais comuns de obter informações de bordas são através de aproximações da primeira e da segunda derivadas, onde valores elevados indicam a existência de bordas no primeiro caso, e no segundo buscam-se mudanças de sinal em f, i.e., cruzamentos em zero. Exemplos clássicos incluem as abordagens de Marr (1982) e o detetor de contornos de Canny (1986), os quais usam aproximações da derivada (de segunda e primeira ordens, respectivamente) juntamente com a convolução com uma função Gaussiana bidimensional, esta última realizando uma função de suavização ou filtragem. Métodos recentes de detecção de contornos são apresentados em Parker (1997).

Por outro lado, técnicas de segmentação baseadas em crescimento de regiões usam um critério de similaridade para agrupar pixels ou regiões a partir de sementes (que para o caso de morfologia matemática aplicada a imagens denominam-se marcadores). A noção de conectividade de regiões R_i é bastante usada, e o objetivo é encontrar um conjunto de regiões $\boldsymbol{R}$, conhecido como partição Ω, que satisfaça critérios como:

1. $\bigcup_{i=1}^{k} R_i = \boldsymbol{R}$.

2. $R_i \cap R_j = \varnothing, \forall\, i, j \in [1, k], i \neq j.$

3. $P(R_i)$ é verdadeiro, $\forall\, i \in [1, k]$.

4. $P(R_i \cup R_j)$ é falso, $\forall\, i, j \in [1, k], i \neq j.$

As condições 1 e 2 definem uma partição convencional de um conjunto, onde no critério 1 assegura-se que todo pixel deve pertencer a uma região, o que implica em uma segmentação completa, onde a união de todas as regiões $\boldsymbol{R}$ é a imagem segmentada. A condição 2 garante que nenhum pixel pertença a mais de uma região ao mesmo tempo enquanto que a condição 3 impõe que cada região R_i da partição Ω satisfaça o predicado lógico P, ou seja, cada R_i satisfaça algum critério definido pelo predicado. Finalmente a condição 4 implica que a união de duas regiões é falsa em relação ao predicado. Neste trabalho assumimos que as regiões encontradas pelo método de segmentação devem ser disjuntas.

O método mais simples de segmentar uma imagem $f(x, y)$ é através da limiarização, onde um valor escolhido entre $[f_{min}, f_{max}]$ é usado para binarizar a imagem. Por exemplo, poderíamos escolher um valor f_x que estivesse em um vale entre dois picos do histograma de f. Uma vez escolhido o valor f_x a operação de limiarização implica em comparar cada pixel de f com f_x, classificando como f_{min} caso $f < f_x$, ou f_{max} caso $f \geq f_x$. Outras técnicas podem usar vários valores limiares, definindo intervalos para limiarização, e ainda podemos usar informações locais de regiões da imagem para detectar valores adequados para os limiares. Como há compressão de informação pelo uso destas técnicas, quase sempre há alguma perda em relação à imagem original f. O objetivo é, então, minimizar as perdas para um dado critério que é, geralmente, específico para o tipo de imagem e sua forma de representação. Em relação a *U-matrix*, o uso de técnicas simples como a limiarização em geral conduz a resultados insatisfatórios, pois geralmente este tipo de imagem possui um histograma complexo e ruidoso, não havendo, em geral, um método simples de encontrar um valor f_x adequado. Uma revisão sobre métodos clássicos de segmentação de imagens é apresentada em Pal e Pal (1993). Parker (1997, capítulo 3) discute vários problemas relacionados à segmentação de imagens em níveis de cinza. A

[3] Uma terceira classe de métodos pode ser definida como os métodos que usam tanto as informações de contorno quanto de crescimento de regiões.

seguir descreve-se o método de segmentação *watershed*, que é usado, neste trabalho, para segmentar a *U-matrix* de forma eficiente.

5.3 O algoritmo *watershed*

Morfologia matemática (MM) é uma teoria geral que estuda a decomposição de operadores entre reticulados completos em termos de algumas famílias de operadores simples (elementares): erosões, dilatações, anti-erosões e anti-dilatações (Barrera et al., 1997). MM é uma ferramenta extremamente poderosa para análise e extração de informações de sinais e imagens. Através de combinações (união, interseção, complemento e composição) dos operadores elementares, podemos formar um grande conjunto de operadores morfológicos. Esta combinação de primitivas pode gerar, então, um operador que satisfaça um objetivo desejado. Apesar de ter sido desenvolvida inicialmente para imagens binárias, os conceitos desenvolvidos em MM têm sido estendidos em vários domínios e a diferentes tipos de imagens. Boas referências a conceitos de MM incluem Matheron (1975), Serra (1982), Giardina e Dougherty (1988), Haralick et al. (1987). Barrera et al. (1994, 1997) descrevem vários operadores morfológicos que foram implementados no *toolbox* MMach, que opera no sistema Khoros (Konstatinides e Rasure, 1994).

Em MM, o principal algoritmo de segmentação de imagens é o algoritmo (ou transformada) *watershed*, proposto inicialmente por Beucher e Lantuejoul (1979), e que tem sido considerado como uma das ferramentas de segmentação mais eficientes em processamento de imagens. Ele pode ser considerado como um algoritmo híbrido, combinando tanto a abordagem de crescimento de regiões quanto detecção de contornos. Uma maneira simples de idealizar o funcionamento do *watershed* é associar a imagem f a um relevo topográfico, por exemplo a figura 5.3, onde considera-se o nível de cinza como altitude. Definimos $B_f(m)$, uma bacia de retenção associada a um mínimo m da superfície topográfica de f, como a região na qual, caso uma gota de água caísse em qualquer ponto desta bacia, iria percorrer um caminho até atingir este ponto de mínimo. A segmentação por *watershed* vai consistir na determinação das bacias de retenção a partir do tipo das primitivas da região e do contorno, a partir dos pontos de mínimos. Assim, podemos considerá-la como um método topográfico de crescimento de regiões. Imaginando gotas de água caindo em todas as coordenadas da imagem, as k bacias de retenção captam a água, partindo de cada mínimo m^k, e à medida que o nível das bacias aumenta é possível que ocorra transbordamento da água de uma bacia para outra. Porém, isto é evitado com a construção de diques, separando as bacias retentoras e impedindo que águas de diferentes bacias sejam compartilhadas. Quando a inundação atinge o nível máximo da altura da superfície, no nosso caso f_{max}, os

diques construídos separando as bacias retentoras são as linhas da *watershed*[4] da imagem, $W(f)$, geralmente com espessura de 1 pixel, formando os contornos das regiões segmentadas da imagem. O *watershed* gera regiões fechadas e conectadas, satisfazendo as condições 1-4 da seção 5.2.

Uma outra maneira de obter as linhas da *watershed*, mais eficiente em termos algorítmicos, é considerar que os mínimos atuem como nascentes de água com diferentes cores, inundando as diferentes bacias de retenção. A mesma interpretação decorreria caso houvesse um furo associado a cada mínimo e submergíssemos a superfície correspondente à imagem f em um lago. Igualmente ao caso anterior, as águas de cada bacia de retenção não devem ser misturadas, sendo construídos diques, que no final do processo, i.e., quando toda a superfície está submersa, representam os contornos entre os objetos segmentados da imagem, i.e., as linhas de *watershed.*

Entretanto, apenas nos casos mais simples pode-se aplicar o *watershed* diretamente na imagem. Dois problemas principais ocorrem na aplicação prática do método: a sobre-segmentação e a sub-segmentação. A primeira leva a imagem final a um grande número de partições, muitas vezes ocasionado por um número grande de mínimos regionais causados por exemplo, por ruído. Por outro lado, a sub-segmentação pode levar à perda de continuidade de algumas "linhas de partições", ocasionando a perda da "bacia", o que no nosso caso implica em uma fusão de dois ou mais agrupamentos de neurônios.

No caso de sobre-segmentação, i.e., um número elevado (acima do desejado) de regiões, o problema ocorre quando a imagem apresenta muitos mínimos regionais, muitas vezes imperceptíveis ao olho humano. Geralmente aplicada sobre o gradiente de uma imagem, uma operação muito sujeita a amplificar ruídos, o uso da *watershed* pode resultar em um número enorme de linhas de partições. O método clássico de regularização usa marcadores específicos, escolhidos de forma a indicar quais as bacias retentoras que são importantes e que devem ser levadas em consideração. Assim, marcadores são regiões onde força-se a existência de uma bacia retentora, i.e., há mudança da homotopia da imagem. O princípio do uso de *watersheds* com marcadores foi proposto por Meyer e Beucher (1990), Meyer (1993), e será usado nesta tese para segmentar a *U-matrix* de um SOM treinado.

Eliminando os marcadores das "bacias" indesejáveis, automaticamente diminuímos o número de partições na imagem final. Apenas as águas de bacias retentoras associadas aos mínimos descritos pelos marcadores terão rótulo, o que implica que o número de regiões finais da imagem segmentada será igual ao número de marcadores escolhidos. A escolha de marcadores para a *U-matrix* será abordada na seção 5.3.2, porém, as figuras 5.5-a e 5.5-b

[4] Também denominadas *linhas de partição de águas.*

ilustram a idéia da escolha de marcadores, usando-se um corte unidimensional em uma dada imagem *f*. Note que na figura 5.5-b apenas alguns marcadores, os relacionados às bacias retentoras mais importantes foram selecionados.

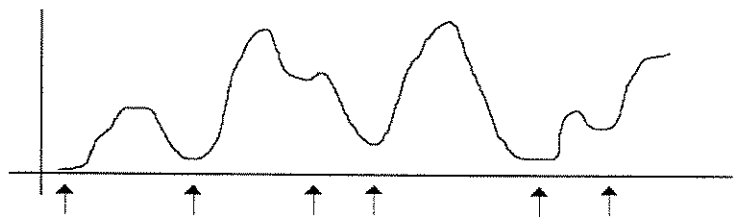


Figura 5.5-a: Marcadores para a *watershed* convencional (todos os mínimos regionais)

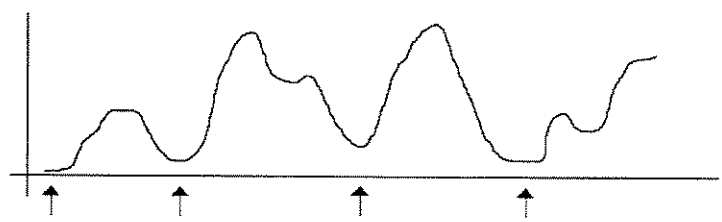


Figura 5.5-b: Marcadores escolhidos para a *watershed* (apenas alguns dos mínimos regionais)

Portanto, a solução seria escolher marcadores antes de iniciar o processo. Os marcadores não necessariamente precisam ser mínimos regionais, bastando apenas que cada marcador, ou marcadores, rotulem uma mesma região, assim, os diques seriam construídos apenas quando a água proveniente de dois rótulos diferentes se encontrarem. Desta forma, uma bacia hidrográfica que não possui marcadores será enchida pela água de uma bacia vizinha (e que primeiro vazar água para dentro desta). Assim, uma das sugestões apresentadas é alterar o algoritmo de forma a aceitar os marcadores nos locais desejados (i.e., forçar a ter os mínimos regionais nos locais indicados). Esta tarefa de determinar os marcadores é denominada de mudança de homotopia da imagem.

É importante notar que o modelo da imagem apresentado na *U-matrix* é adequado ao tratamento com o *watershed*, onde os vales caracterizando os agrupamentos dos neurônios em partes do mapa são associados a bacias de retenção de águas. Busca-se aqui um método eficiente de determinar tanto o número de marcadores, suas posições, e as regiões derivadas do processo de crescimento topográfico a partir destes, efetuado pela *watershed*.

Um modo mais formal de definir a *watershed* foi proposto por Meyer (1993) usando o conceito dos menores caminhos (*shortest paths*). Descrevemos esta idéia de forma bastante sucinta, baseada em Meyer (1993) e Meijster e Roerdink (1996), a seguir.

5.3.1 Cálculo do *watershed*

Considere uma imagem digital em níveis de cinza como uma função $f : D \rightarrow \mathbb{N}$, onde $D \subseteq \mathbb{Z}^2$ é o domínio da imagem e $f(p)$ o nível de cinza do pixel $p \in D$. Seja $E \subset \mathbb{Z}^2 \times \mathbb{Z}^2$ o conjunto de coordenadas da imagem, P o caminho entre dois pixels p e q e com comprimento $l(P)$, ver seção 5.2. Para um pixel $p \in D$, seja $N_E(p) = \{ q \in D \mid (p, q) \in E \}$ o conjunto de pixels vizinhos de p.

O custo de ir de uma posição p a uma posição vizinha q pode ser definido como

$$custo(p,q) = \begin{cases} LS(p), & \text{caso } f(p) > f(q) \\ LS(q), & \text{caso } f(p) < f(q) \\ \dfrac{LS(p) + LS(q)}{2}, & \text{caso } f(p) = f(q) \end{cases}$$

onde $LS(p)$ é a rampa máxima ligando um pixel p a qualquer um de seus vizinhos de menor altitude, definida como

$$LS(p) = \underset{q \in \{p\} \cup N_E(p)}{\text{MAX}} [f(p) - f(q)]$$

A distância topográfica entre dois pontos p e q ao longo de um caminho $P = (p_0, \ldots, p_{l(P)})$ é definida como

$$T_f^P(p,q) = \sum_{i=0}^{l(P)-1} custo(p_i, p_{i+1})$$

e a distância topográfica entre dois pontos p e q é definida como a distância topográfica mínima considerando todos os caminhos entre p e q:

$$T_f(p,q) = \underset{P \in p \mapsto q}{\text{MIN}} T_f^P(p,q)$$

onde $p \mapsto q$ representa o conjunto de todos os caminhos de p a q. Note que $T_f(p, q)$ é nula caso os pixels p e q sejam do mesmo objeto e possuam o mesmo nível de cinza. A distância topográfica entre um ponto $p \in D$ e um conjunto $A \subseteq D$ é definida como

$$T_f(p,A) = \underset{a \in A}{\text{MIN}} T_f(p,a).$$

Seja a função f^* definida como a função f, porém substituindo todos os valores dos mínimos locais por 0, ou seja, caso p seja um mínimo $f^*(p) = 0$, caso contrário, $f^*(p) = f(p)$. Seja $(m_i)_{i \in I}$ o conjunto de mínimos da função f^*. A base de retenção de um mínimo m_i, denotada por $CB(m_i)$, é definida como o conjunto de pontos $p \in D$ que estejam topograficamente mais próximos a m_i do que a qualquer outro mínimo m_j:

$$CB(m_i) = \left\{ p \in D \mid \forall j \in I \setminus \{i\} : T_{f^*}(p,m_i) < T_{f^*}(p,m_j) \right\}$$

e o *watershed* de uma função f é o conjunto de pontos deste domínio que não pertence a nenhuma bacia de retenção

$$W(f) = D \cap \left[\bigcup_{i \in I} CB(m_i) \right]^c.$$

onde c denota o conjunto complemento.

Métodos de computação de *watershed* baseados em algoritmos de grafos paralelos, usando o algoritmo de Dijkstra (1959), foram apresentados em Meijster & Roerdink (1996). Outro modo de implementação é usando o conceito de filas hierárquicas, onde há uma fila para cada nível de cinza e um pixel é processado a cada vez (Noguet *et al.* (1996), (Facon, 1996). Exemplo prático da aplicação da *watershed* em segmentação de imagens citológicas foi apresentado em Costa et al. (1997c).

5.3.2 Escolha dos marcadores

Através do paradigma de Meyer e Beucher (1990), o problema de detecção de contornos, em geral bastante complexo, é substituído por um problema mais simples que é o de achar marcadores para os objetos de interesse na imagem. Porém, não existe forma geral de encontrar os marcadores, sendo, em alguns casos, de grande importância o conhecimento *a priori* do usuário acerca da imagem dos objetos a serem segmentados. Em alguns casos, é possível que o usuário, através de uma interface, escolha adequadamente as posições dos marcadores, e deixe que o algoritmo *watershed* se encarregue de detectar, de forma ótima, as bordas entre os objetos.

Apesar de termos testado alguns métodos para escolha de bons marcadores para a *U-matrix*, um método relativamente simples tem gerado bons resultados em uma grande diversidade de problemas. Seja a *U-matrix* de um SOM treinado dada pela imagem f, de tamanho $2N\text{-}1 \times 2M\text{-}1$, onde $N \times M$ é o tamanho do mapa. Considere que $[f_{min}, f_{max}] = [0, 255]$ e n_i é 1, ou seja, há 256 níveis de cinza na imagem f. Os seguintes passos são efetuados:

1. Filtragem: a imagem f_1 é gerada removendo-se pequenos buracos na imagem f. Pequenas depressões com área inferior a τ pixels são eliminadas.

2. Para $k = 1, \ldots, f_{max}$, onde f_{max} é o nível de cinza máximo na imagem f_1, crie as imagens binárias f_2^k correspondendo a conversões de f_1 usando k como valor de limiar.

3. Calcule o número de regiões conectadas de f_2^k, para cada valor de k, N_{rc}^k.

4. Procure no gráfico $k \times N_{rc}^k$ a maior seqüência contígua e constante de número de regiões conectadas N_{rc}^k, denotado por S_{max}.

5. A imagem de marcadores será a imagem f_2^j, onde j é o valor inicial[5] da seqüência S_{max}.

O passo 1 suaviza, muito discretamente, a imagem original, resultando em uma imagem melhor para processamento, visto que a *U-matrix* possui, em geral, muitas rugosidades. O valor de τ utilizado foi 3, para todas as experiências. A operação de filtragem poderia também ser implementada a partir de filtros morfológicos ou erosões seguidas de dilatações, usando um elemento estruturante de raio ρ. No caso, quanto maior ρ mais forte será a filtragem. O objetivo de eliminar pequenas depressões com área inferior a 3 pixels praticamente não altera, visualmente, a imagem. Porém, caso não a façamos, a operação de limiarização (passo 2) irá produzir vários objetos pouco significativos, i.e., objetos com 1 pixel apenas, que no nosso caso, em geral, não são importantes.

O passo 2 pode ser visto como se pudéssemos fatiar a imagem em todos os seus níveis de cinza, do mínimo ao máximo, usando estes níveis como limiares para binarizar a imagem. Para cada imagem binária derivada de uma operação de limiarização, a operação de rotulação atribui um código diferente a cada um dos componentes conectados, resultando

[5] Apesar de podermos utilizar qualquer valor na seqüência S_{max}, o valor inicial gera marcadores com menor área, fazendo com que o método *watershed* descubra as linhas de partições de forma mais adequada. Caso tivéssemos escolhido o valor máximo em S_{max}, teríamos poucos níveis de cinza para a dinâmica do algoritmo *watershed*.

em um número de objetos na imagem para cada valor limiar k aplicado, N_{rc}^{k}. A figura 5.6 ilustra, para a U-matrix apresentada na figura 5.3, o gráfico do valor do limiar k versus N_{rc}^{k}.

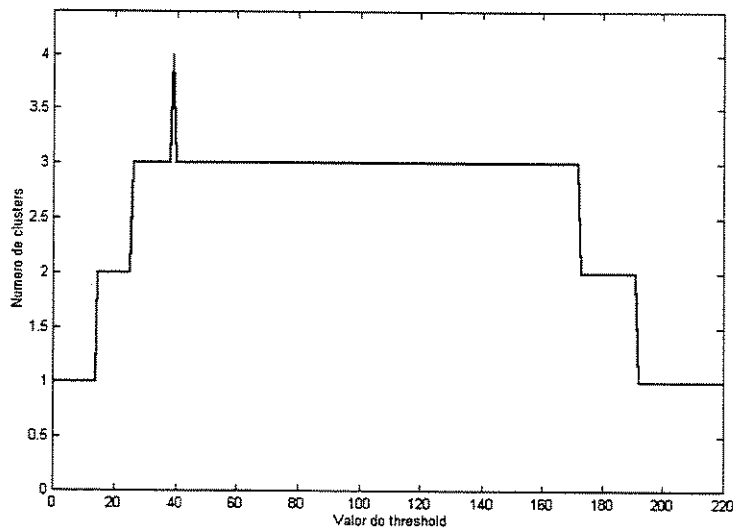


Figura 5.6: Gráfico do valor do limiar k versus N_{rc}^{k}, para a U-matrix apresentada na figura 5.3.

Como era esperado, pela análise visual da figura 5.3, temos uma grande estabilidade a partir de $k = 43$ e indo até $k = 170$, para N_{rc} igual a 3 marcadores. Esta seqüência contígua e constante do mesmo valor nos permite escolher $\psi = 3$ como o número de marcadores da imagem f. A escolha de qual valor k deve ser usado, uma vez sabendo a seqüência mais estável, foi feita de forma a pegar o início da seqüência, i.e., a imagem de marcadores é a imagem binária obtida pela limiarização de f usando como valor limiar j, sendo que j é o primeiro valor da seqüência mais estável de número de regiões conectadas quando aplica-se os múltiplos limiares na imagem.

A figura 5.7 ilustra os marcadores m_i, $i = 1, \ldots, 3$, obtidos para o caso da figura 5.3. Aplicando-se a *watershed* sobre a imagem f, usando os marcadores m_i's, obtém-se as linhas de *watersheds* $W(f)$ como mostrado na figura 5.8. Note que a conectividade usada para o pixel p é $N_8(p)$.

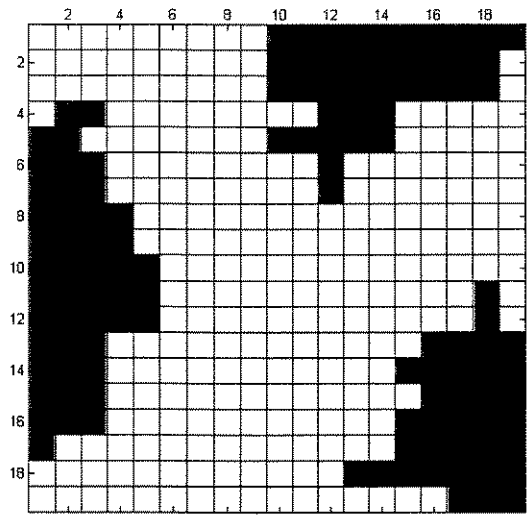


Figura 5.7: Marcadores obtidos para a U-matrix da figura 5.3 usando como limiar k = 43.

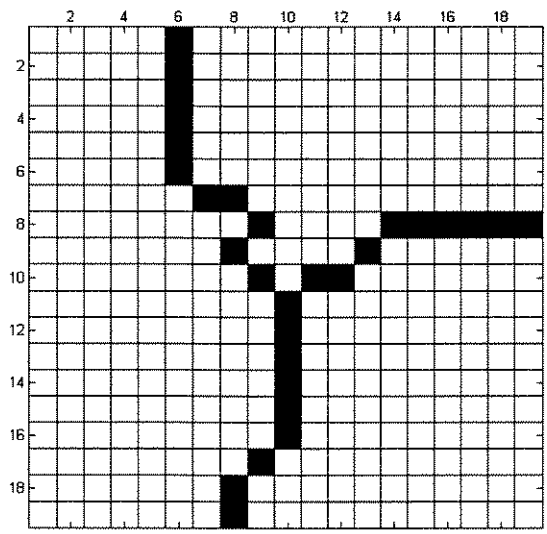


Figura 5.8: Linhas da watershed obtidas da U-matrix da figura 5.3, usando os marcadores apresentados na figura 5.7.

As figuras 5.9 e 5.10 ilustram a sobreposição das linhas de *watershed* sobre a U-matrix original, para os casos bidimensional e tridimensional. Note que as bordas separando as regiões são agora todas fortes[6], i.e., não há mais ambigüidades, e as linhas de *watershed* representam os contornos ótimos entre as regiões, dado os marcadores.

[6] A informação de bordas fracas e bordas fortes será usada em trabalhos futuros onde haverá uma análise *fuzzy* das regiões de neurônios da *U-matrix*.

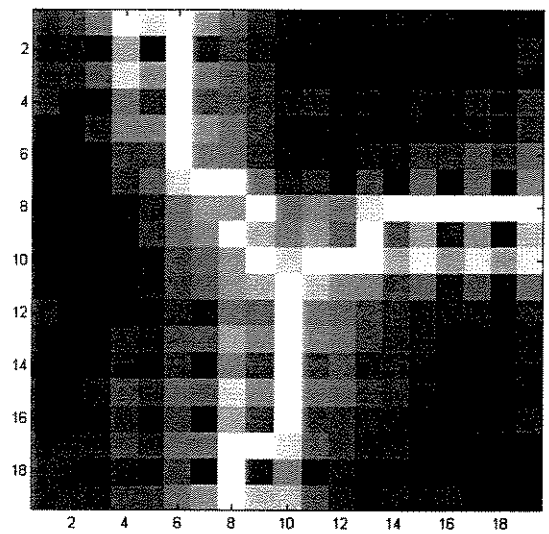


Figura 5.9: Linhas de watershed sobrepostas à U-matrix original, figura 5.3, em 2D.

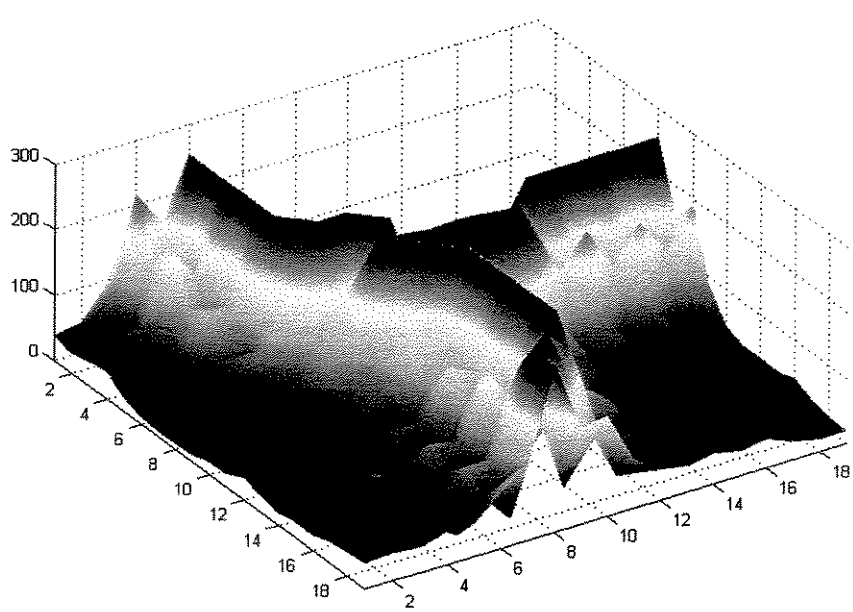


Figura 5.10: Linhas de watershed sobrepostas à U-matrix original, figura 5.3, em 3D.

5.3.3 Rotulagem de regiões conectadas

A extração e rotulação dos vários componentes conectados e disjuntos da imagem são tarefas centrais em muitos dos sistemas de análise automática de imagens. No nosso caso, a imagem binária resultante da aplicação do *watershed*, (ver figura 5.8) deve ser rotulada de forma que possamos tratar os vários objetos de forma independente. Considere que a imagem binária possui os valores 0 para o fundo da imagem e 255 para os objetos conectados. O algoritmo *rotulação de componentes conectados* (RCC) percorre a imagem binária, por exemplo do canto superior esquerdo ao canto inferior direito, atribuindo um código (ex. um número) a cada região ou seqüência de pixels adjacentes, de acordo com o padrão de conectividade desejado (ver seção 5.2).

Parte-se de um valor *código_atual* que é atribuído ao primeiro pixel com nível de cinza 255, i.e., o pixel faz parte de uma região em *foreground* da imagem. A este pixel denominamos semente. Todos os vizinhos da semente que possuam nível de cinza 255 são rotulados e igualmente passam a ser sementes, o que recursivamente faz preencher toda uma região conectada da imagem. Regiões diferentes recebem códigos diferentes fazendo com que caso um pixel seja detectado com valor 255 e não possua nenhum vizinho com nível de cinza diferente de 0 ou 255, o *código_atual* seja incrementado de um valor, por exemplo 1. Sempre que detecta-se um pixel p em *foreground*, analisa-se a vizinhança $N_m(p)$, onde m dita a conectividade escolhida. Caso já houver um código atribuído a algum pixel $q \in N_m(p)$, este código é usado para rotular o pixel p.

Assim, todas as regiões e pixels adjacentes, de acordo com o padrão de conectividade escolhido, recebem o mesmo valor de código, e assim podem ser tratadas independentemente por rotinas de análise de imagem, referenciando o código como um dos parâmetros na chamada das funções. Maiores detalhes e o pseudo-código de RCC pode ser visto em Costa (1996a). Um exemplo de rotulação é mostrado na figura 5.11, onde foi efetuado o RCC sobre a imagem binária apresentada na figura 5.8.

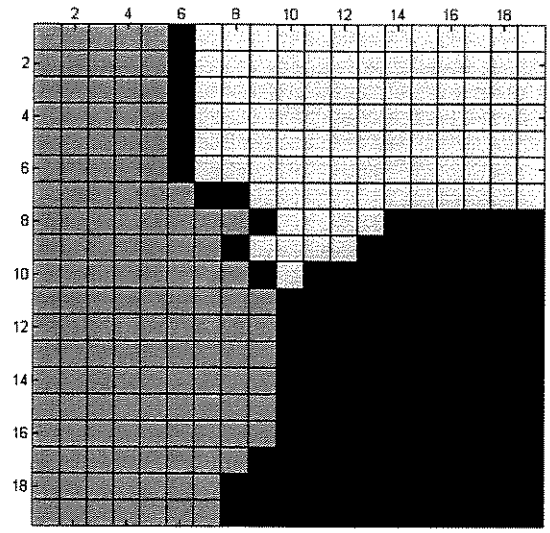


Figura 5.11: Regiões conectadas da U-matrix rotuladas pelo algoritmo RCC.

5.4 O algoritmo SL-SOM

Toda análise e segmentação de um mapa de tamanho $N \times M$ treinado é feita inicialmente sobre a *U-matrix*, e posteriormente as informações associadas aos pixels são associadas aos neurônios.

Os passos do SL-SOM (de *Self-Labeled SOM*), a partir do treinamento, e considerando que tenhamos obtido sucesso, i.e., uma boa ordenação, são descritos a seguir.

Passos do algoritmo SL-SOM:

1. Obtenção da *U-matrix* (seção 5.1)

2. Encontrar os marcadores para a *U-matrix* (seção 5.3.2)

3. Aplicar o *watershed* sobre a *U-matrix* usando os marcadores obtidos no passo 2.

4. Rotulagem das regiões conectadas da imagem segmentada no passo 3 (seção 5.3.3).

5. Cópia dos rótulos obtidos no passo 4 para os neurônios associados a cada pixel da *U-matrix*.

6. Caso ainda existam neurônios não rotulados, o que pode ocorrer caso o pixel associado da *U-matrix* faça parte das linhas de *watershed*, rotule-os usando o método vizinho mais próximo[7], calculando as distâncias no espaço de pesos dos neurônios, e atribuindo o código do neurônio rotulado mais próximo.

Exemplos para os passos 1-4 foram mostrados. A figura 5.12 ilustra o resultado para o passo 5. Note que três neurônios permaneceram sem código, por exemplo o neurônio (4,4). Aplicando a estratégia descrita no passo 6, obtêm-se o SOM totalmente rotulado, o que é mostrado na figura 5.13.

Por outro lado, poderíamos deixar os neurônios não rotulados como está mostrado na figura 5.12 e manter atenção apenas nos neurônios rotulados até o passo 5. Caso um padrão $\boldsymbol{x}$ seja mapeado no neurônio não rotulado, poderíamos, neste caso, buscar o neurônio rotulado mais próximo do padrão $\boldsymbol{x}$, ou os K-vizinhos mais próximos. Isto poderia ser feito, por exemplo, buscando o segundo neurônio vencedor para o padrão $\boldsymbol{x}$, e caso este também estivesse não rotulado, o terceiro, e assim por diante. A classe, ou código atribuído pelo algoritmo RCC, do neurônio rotulado encontrado, nesta seqüência, é usado para rotular o padrão $\boldsymbol{x}$.

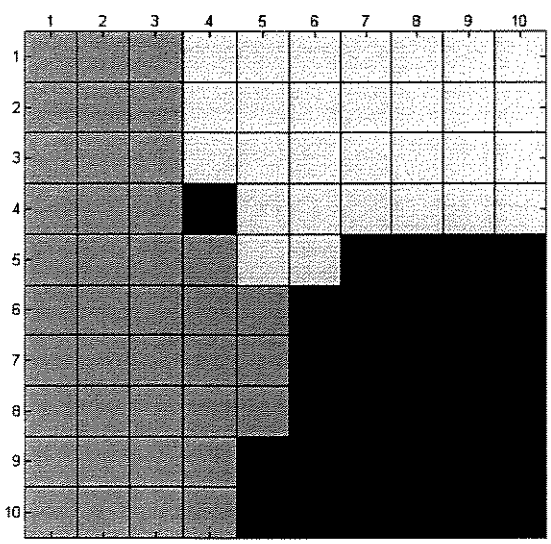


Figura 5.12: Rotulagem dos neurônios do SOM 10×10 pela cópia dos códigos da U-matrix rotulada (figura 5.11)

[7] Obviamente, pode-se usar também o método K-vizinhos mais próximos, buscando qual classe possui K elementos mais próximos, no espaço de pesos, do neurônio não rotulado.

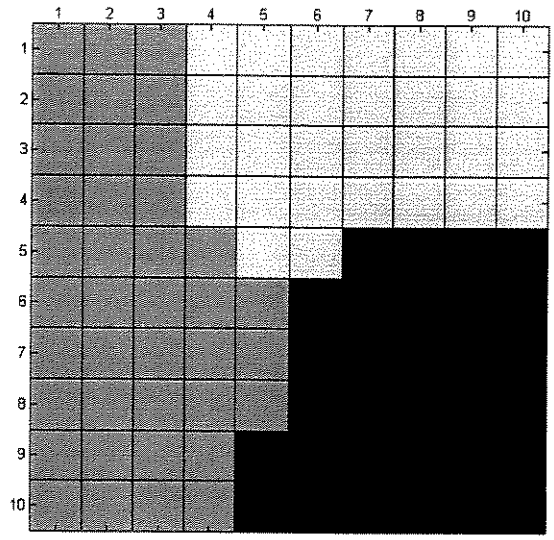


Figura 5.13: Mapa 10×10 totalmente rotulado para o problema apresentado na seção 3.6, usando o método vizinhos mais próximos para classificar os pixels não rotulados da figura 5.12.

A figura 5.14 ilustra o mapeamento dos padrões no mapa 10×10. A forma convencional de rotular um mapa treinado é fazer com que cada padrão seja mapeado no SOM, i.e., seja apresentado e detectado o neurônio vencedor. A classe atribuída ao neurônio é a classe dos padrões mais freqüentes que são mapeados nele. Desta forma, usa-se o SOM apenas para fazer uma projeção dos dados de um espaço p-dimensional para um *grid* bidimensional, e o processo de rotulação usa informação *a priori*, o que em muitos casos não está disponível. Por exemplo, o neurônio (1,1), que está no canto superior esquerdo do mapa, foi rotulado como pertencente à classe 2. Note que nem todos os neurônios receberam rótulos, o que pode ser visto na figura 3.21, onde o histograma de vencedores para este problema é apresentado. Alguns neurônios não estão mapeando nenhum padrão, e desta forma, no método convencional não há, em princípio, como classificá-los. Outro problema freqüente é o conflito, o que ocorre quando dois padrões de classes distintas são mapeadas no mesmo neurônio. Geralmente ocorre quando as classes possuem algum nível de sobreposição, quando os dados estão mal condicionados, ou quando escolhemos de forma errada os atributos usados no processo. Pode-se, quando possível, buscar encontrar atributos adicionais que habilitem a separabilidade das classes.

De posse da figura 5.14, pesquisadores efetuam manualmente uma separação de regiões, em geral não considerando os neurônios não rotulados. Vê-se que o número de agrupamentos, no método tradicional, deve ser conhecido a priori, caso contrário, não há como segmentar manualmente o mapa. Exemplos de rotulação manual são apresentados, por exemplo, em Zupan e Gasteiger (1993) e Kohonen (1997a).

2	2		1	1	1	1	1	1	1
2	2	2	1	1	1	1	1	1	1
2	2	2	1	1	1	1	1	1	1
2	2	2		1	1	1	1	1	1
2	2	2		1	1				3
2	2		2	2	3	3	3	3	3
2	2	2	2	2	3	3	3	3	3
2	2	2	2	3	3	3	3	3	3
2	2	2	2		3	3	3	3	3
2	2	2	2	3	3	3	3	3	3

Figura 5.14: Mapeamento dos padrões no mapa 10×10 usando a informação a priori das classes dos padrões.

O método SL-SOM efetuou, automaticamente, segmentação e rotulação do SOM (ver figura 5.13) apenas usando os padrões. Nenhuma informação de classe foi usada. A figura 5.15 ilustra a sobreposição da figura 5.14 em 5.13. Vemos que, neste problema, a segmentação foi perfeita, i.e., nenhuma padrão de uma classe foi mapeado em uma outra classe. Outra informação importante é que o método determinou o número de agrupamentos nos dados, 3, sem auxílio de informações privilegiadas, como ocorre no caso convencional.

Também importante, o SL-SOM gerou conjuntos de neurônios sob o mesmo rótulo, o que corresponde à representação distribuída dos protótipos dos agrupamentos como foi discutida no capítulo 2. Cada agrupamento agora é descrito por um conjunto de neurônios, cada neurônio um vetor no espaço p-dimensional. A figura 5.16 ilustra isto, onde cada neurônio recebeu um código de acordo com o agrupamento que pertence. Note que todos os neurônios foram representados de forma similar em relação ao tamanho, mudando apenas o nível de cinza, que corresponde ao código do agrupamento. A figura 5.17 usa o histograma de vencimentos, i.e., quantos padrões cada neurônio está mapeando para representar o tamanho do neurônio no *grid*. Os números sobre os neurônios (1, 2 e 3) são os códigos das classes descobertas durante o processo de rotulação, i.e., RRC. Assim, a classe de padrões 1, que é a que está mais a esquerda da figura 3.9 é representada pelo agrupamento 2, descoberto pelo SL-SOM. A classe de padrões 2 é representada pelo agrupamento rotulado como 1, e na classe de padrões 3, o código atribuído ao agrupamento pelo método RRC foi 3. Não há nenhuma preocupação em coincidência do rótulo do agrupamento com o rótulo da classe dos padrões, mesmo porque o SL-SOM não usou em nenhuma fase informações da classe, e os códigos atribuídos pelo método RRC seguem uma numeração automática, a medida que vai encontrando regiões não rotuladas na imagem. O importante é que o

método descobriu a existência das classes, com número certo, e gerou um agrupamento de neurônios através da segmentação e rotulação de forma que cada agrupamento de neurônios corresponde a um modelo dos dados, que pode ser usado, inclusive para classificar novos padrões não usados na fase de treinamento.

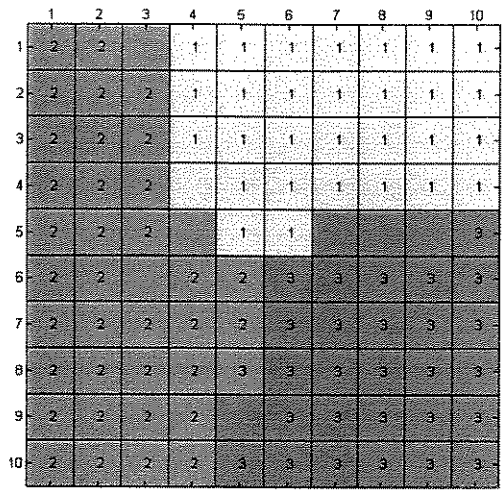


Figura 5.15: Mapeamento dos padrões no mapa 10 × 10, usando a informação das classes dos padrões, sobreposto à partição do mapa obtida automaticamente.

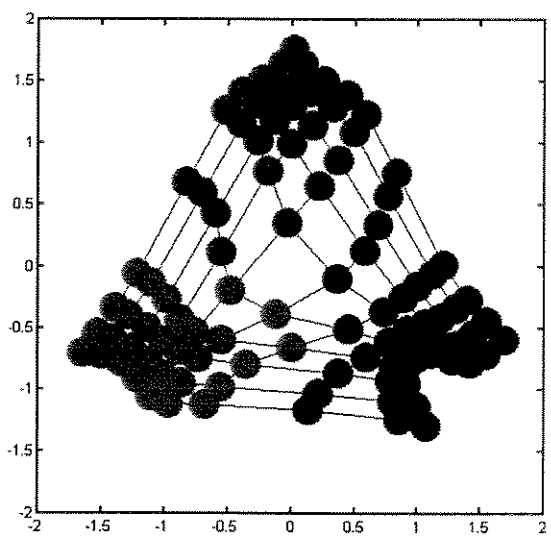


Figura 5.16: Grid do mapa 10 × 10 rotulado. Na figura todos os neurônios foram representados com o mesmo tamanho, diferenciando apenas o nível de cinza, que está relacionado ao agrupamento detectado.

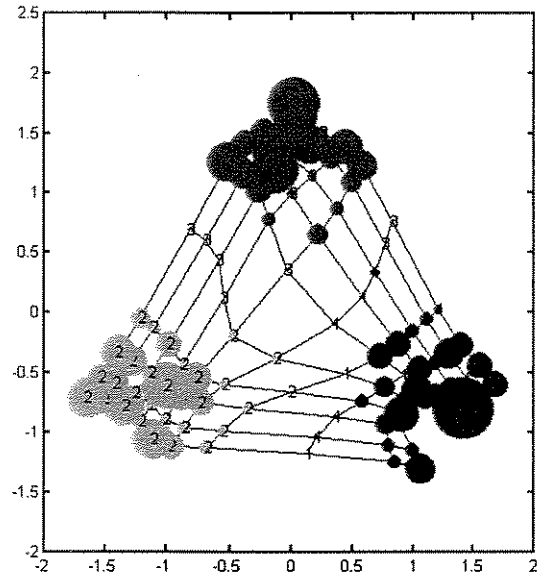


Figura 5.17: Grid do mapa 10 × 10 rotulado, usando a informação de número de padrões mapeados em cada neurônio para representar o tamanho no grid.

Uma vez dispondo do SOM rotulado, como apresentado na figura 5.15, os dados são classificados de acordo com a classe do agrupamento de neurônios onde se encontra o neurônio o que está mapeando. A figura 5.18 ilustra a classificação dos dados obtida através deste método. Partindo apenas do conjunto de dados, ver tabela 2.1, o processo de classificação automática pode ser pensado como um método que adiciona uma coluna à matriz de dados, onde esta coluna é exatamente a classe do padrão.

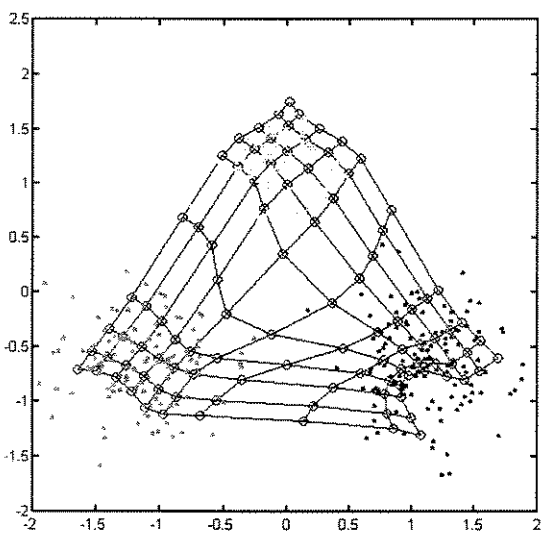


Figura 5.18: Classificação dos padrões usando a classe dos neurônios vencedores.

A seguir apresentamos uma série de exemplos de conjuntos de dados, os quais ilustram a adequação do método a uma variedade de situações. Todas as simulações foram efetuadas no ambiente Matlab 5.0.

5.5 Exemplos de aplicação e análise do SL-SOM em alguns conjuntos de dados

5.5.1 O conjunto de dados *chainlink*

Um exemplo não trivial para comparações de métodos de agrupamentos em dados multidimensionais foi proposto por Ultsch (1995), que é o *chainlink*. O conjunto de dados *chainlink* consiste de 1000 pontos no espaço $\Re^3$ tal que eles possuem a forma de dois anéis tridimensionais entrelaçados. Um dos anéis se estende na direção *x-y* enquanto o outro se estende na direção de *x-z*. Os dois anéis podem ser pensados como elementos de uma corrente, cada um consistindo de 500 objetos de dados.

Este problema ilustra a capacidades do SOM em descobrir a estrutura dos dados mesmo para conjuntos de dados com forma complexas e não-esféricas, e não separáveis linearmente. Alguns destes dados foram apresentados em Costa e Netto (1999b). A Figura 5.19 ilustra o conjunto de dados usado. Efetuando análises estatísticas junto ao conjunto de dados, pode-se descobrir algumas informações sobre a distribuição dos dados. Usando análise de componentes principais (PCA) pode-se ver que as três variáveis são, de fato, descorrelacionadas.

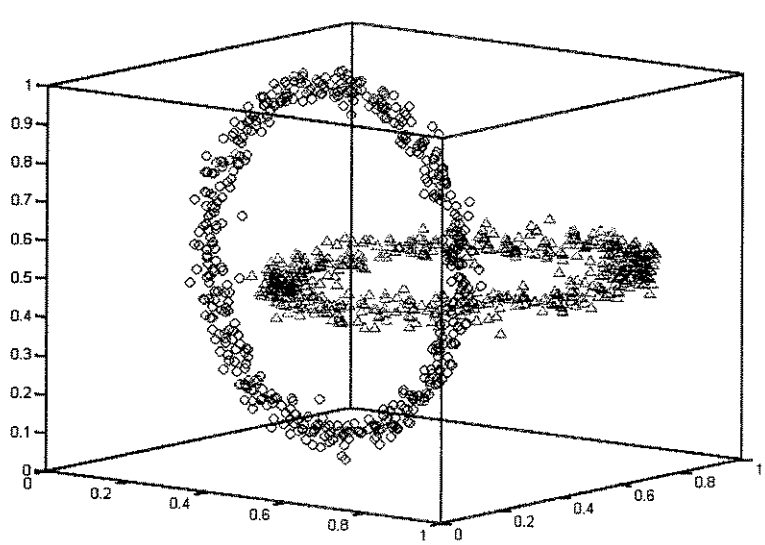


Figura 5.19 - O conjunto de dados chainlink

Como descrito no capítulo 2, a projeção dos dados via PCA poderia nos indicar se os dados podem ou não ser representados em um número reduzido de dimensões. A operação consiste em projetar os dados em sub-espaços vetoriais, ortogonais, que preservem ao máximo a variância do conjunto de dados. O primeiro componente resume os dados com a máxima variância quanto possível em uma direção, seguido pelo segundo componente, que possui direção ortogonal à primeira, e assim por diante. Nos casos onde há possibilidade de redução de dimensionalidade, os últimos componentes possuem variância desprezível, e em geral podemos escolher apenas um conjunto reduzido de componentes (os primeiras k componentes), que absorvem a máxima variabilidade dos dados. Nestes casos, em geral, o descarte dos últimos componentes não interferem signicativamente no resultado final. Porém, observando a tabela 5.2, vemos que não dá para encontrar um sub-espaço com duas dimensões para este conjunto de dados. Cada variável é responsável por cerca de um terço da variância total. Desta forma, o conjunto de dados deve ser representado pela mesma dimensão original, de acordo com a PCA[8]. A figura 5.20 ilustra distribuições dos dados ao longo de projeções tomando-se pares de variáveis, mostrando o conjunto de dados a partir de diferentes pontos.

TABELA 5.2 - PERCENTUAL DA VARIÂNCIA TOTAL EXPLICADA PELA ANÁLISE USANDO PCA

	1a. Comp.	2a. Comp.	3a. Comp.
% Explic.	33,5%	33,3%	33,2%

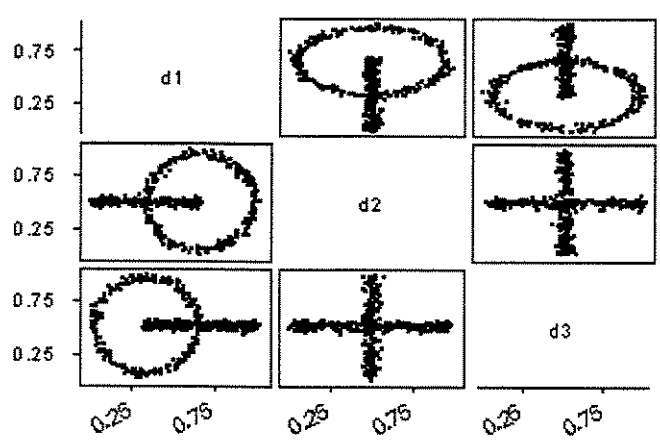


Figura 5.20: Distribuições dos dados ao longo das projeções tomando-se pares de variáveis

[8] A análise efetuada foi sobre os autovetores e autovalores da matriz de correlação.

Assim, o conjunto de dados não é redutível a duas dimensões, nem tampouco as classes são linearmente separáveis. O objetivo é, partindo apenas do conjunto de dados, $\boldsymbol{X}$, determinar o número de classes presentes nos dados, e prover conjuntos de neurônios no modelo distribuído de protótipos para cada classe, caso sejam descobertas.

O mapa usado nesta experiência tem tamanho 15×15. A inicialização de pesos foi linear e o treinamento foi efetuado com o algoritmo de atualização em lote (*batch*), ver capítulo 3. A função de vizinhança usada foi Gaussiana e o raio inicial foi 12, caindo para 1 de forma linear com o tempo. O número de épocas foi fixado em 500. A figura 5.21 ilustra a configuração dos neurônios no espaço 3-D após o final do treinamento. A relação de vizinhança é expressa por linhas que conectam os neurônios.

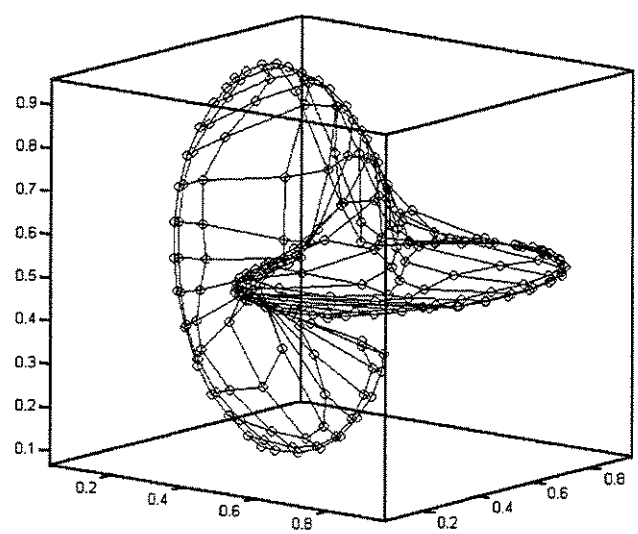


Figura 5.21 - Grid do SOM 15×15 após 500 iterações usando o algoritmo de atualização em lote

A *U-matrix* correspondente à figura 5.21 é apresentada na figura 5.22. Embora a *U-matrix* seja um método de visualização do relacionamento entre os neurônios, a automatização de descoberta da conhecimento pela *U-matrix* não é direta, principalmente em casos onde as bordas não estejam bem delimitadas. O uso de *watershed* é motivado pela busca do contorno ideal separando as regiões de neurônios que representam os agrupamentos, dados os marcadores, que podem ser vistos como os núcleos dos agrupamentos.

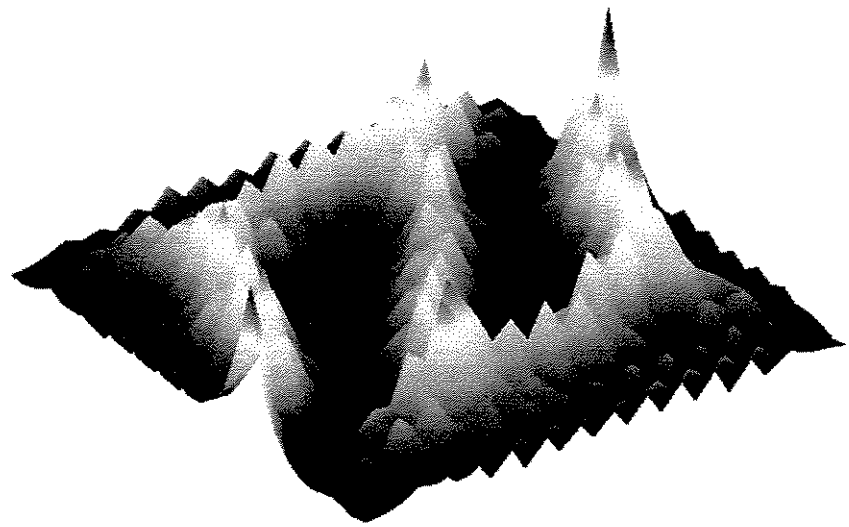


Figura 5.22 - U-matrix para a configuração de neurônios apresentada na figura 5.21

A figura 5.23 mostra o número de regiões conectadas (N_{cr}^{k}) para cada valor de limiar da *U-matrix*. O algoritmo decidiu automaticamente o número correto de agrupamentos (2). De acordo com o algoritmo SL-SOM, a imagem de marcadores correspondeu à imagem binária obtida da limiarização da *U-matrix* com o valor inicial da seqüência mais estável de N_{cr}, que no caso foi $k = 24$. A figura 5.24 mostra as linhas de watershed sobrepostas à *U-matrix* apresentada na figura 5.22. A partição da *U-matrix* (já rotulada) é apresentada na figura 5.25, onde os dois agrupamentos são mostrados separados pelas linhas de *watershed* (em preto).

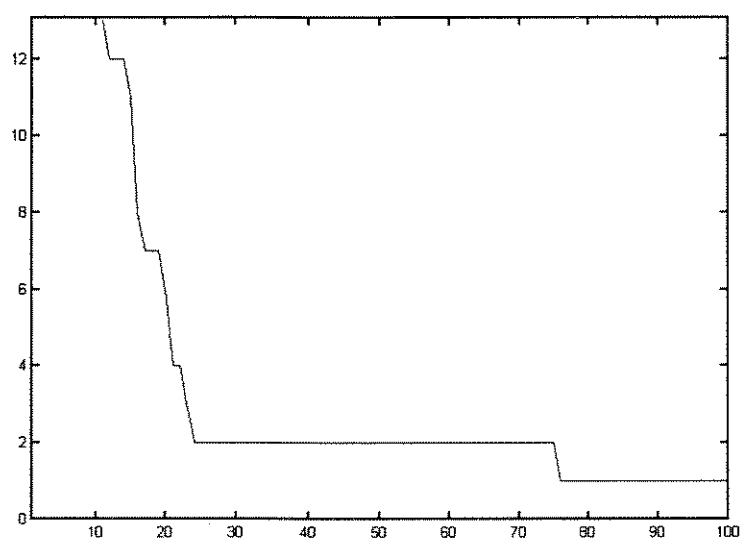


Figura 5.23: Gráfico do número de regiões conectadas (N_{cr}^{k}) para cada valor de limiar, k, da *U-matrix*.

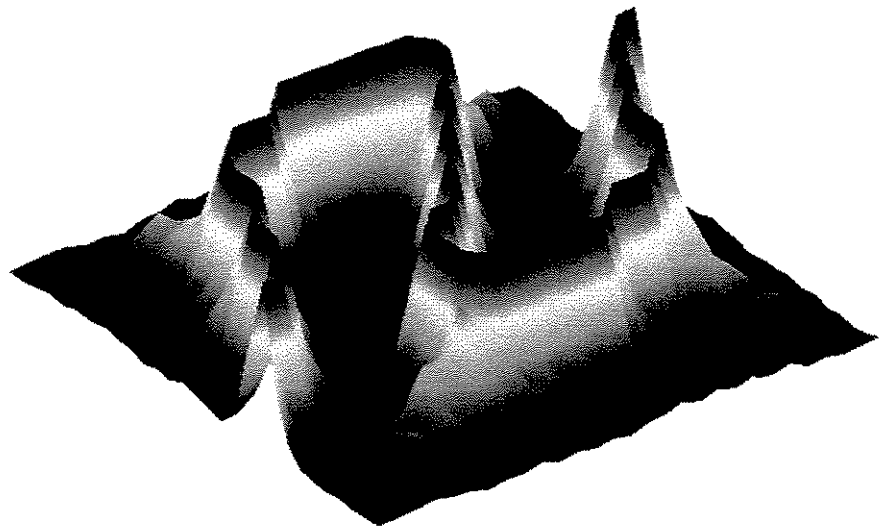


Figura 5.24: Linhas de watershed sobrepostas à U-matrix apresentada na figura 5.22.

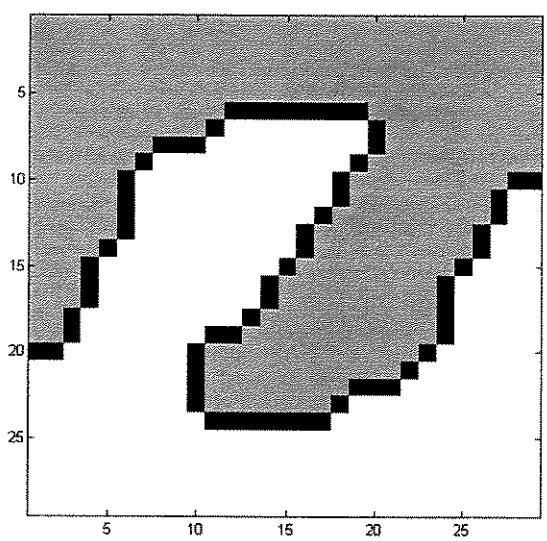


Figura 5.25: Partição da U-matrix (já rotulada) onde os dois agrupamentos são mostrados separados pelas linhas de watershed (em preto).

Rotulando os neurônios como mostrado na seção 5.4 e mapeando todos os padrões no SOM, podemos analisar o resultado da partição obtida. Neste exemplo, todos os padrões foram classificados corretamente pelo método apresentado. A figura 5.26 ilustra o formato dos protótipos distribuídos para cada agrupamento. Note que foi usado um raio de influência de tamanho 0.1 em cada neurônio, porém, como discutimos no capítulo 2, a

influência de cada neurônio constituinte de um modelo distribuído estende-se até o infinito, a menos que estabeleçamos um limiar (no caso de desejarmos evitar classificação de valores discrepantes), ou até que as influências de dois ou mais neurônios se toquem, o que delimita o espaço ou região de influência de um neurônio. No caso da figura 5.26, apenas os neurônios ativos foram usados, i.e., neurônios que não obtiveram nenhum padrão em sua região de influência foram ignorados, para efeito de geração da figura.

Note que, apesar de considerar as influências de cada neurônio no espaço p-dimensional como isotrópicas, a influência no espaço do conjunto de neurônios rotulados como um só agrupamento, que é o hipervolume de todas as influências dos neurônios que fazem parte do conjunto, pode ter geometria livre. Esta é uma grande vantagem em relação ao SOM tradicional, onde usa-se, por exemplo, k neurônios em um *grid* unidimensional para efetuar agrupamentos, onde geralmente sabe-se a priori que k é o número correto de classes. A mesma coisa ocorre por exemplo com o método *k-means*, o qual busca no espaço, via otimização numérica, k centros que são os protótipos das classes. O *k-means* sempre busca agrupamentos hiper-esféricos, e o resultado da influência de cada protótipo no espaço é similar ao mostrado na figura 2.7. O resultado da classificação do *k-means* é mostrado na figura 5.27, onde são mostradas também os dois centros ou protótipos encontrados. Note que houve inúmeros erros e a estrutura correta dos agrupamentos não foi obtida, mesmo usando a informação privilegiada do número de agrupamentos, 2.

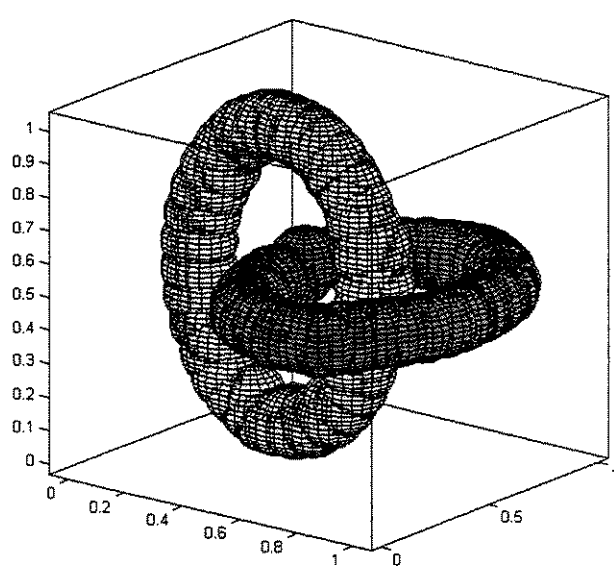


Figura 5.26: Geometria dos agrupamentos descobertos usando o modelo distribuído de protótipos - A influência de cada neurônio foi limitada, para efeito de geração da figura, em 0.1

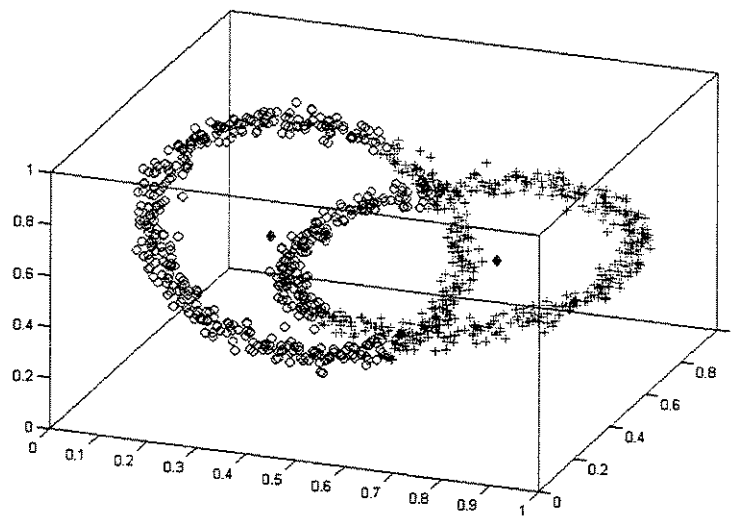


Figura 5.27 - Resultado do k-means para o conjunto de dados chainlink com k = 2.

Usando outros valores para k o problema continua a não ser resolvido pelo *k-means*. Por exemplo, os resultados para $k = 3$ e $k = 6$ são apresentados nas figuras 5.28 e 5.29. Comparando os resultados do *k-means* com o obtido pelo SL-SOM vemos que este último conseguiu recuperar e representar fielmente a estrutura dos dados, mesmo neste problema onde as classes não são linearmente separáveis.

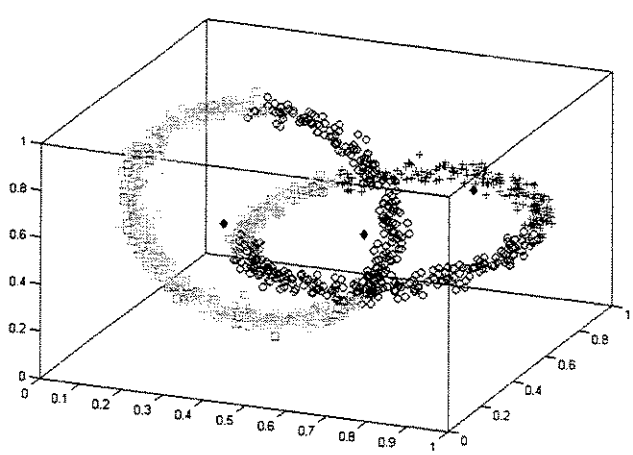


Figura 5.28 - Resultado do k-means para o conjunto de dados chainlink com k = 3.

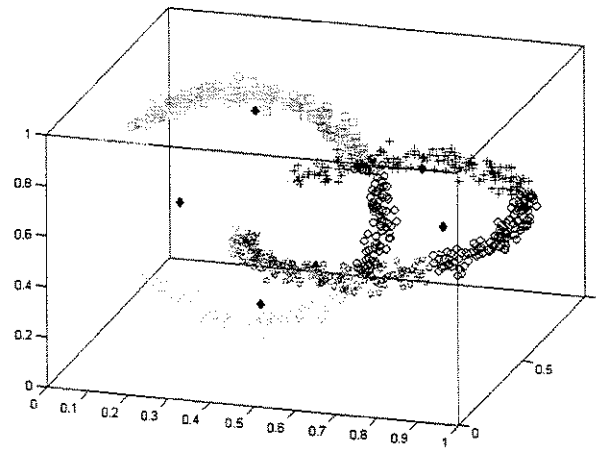
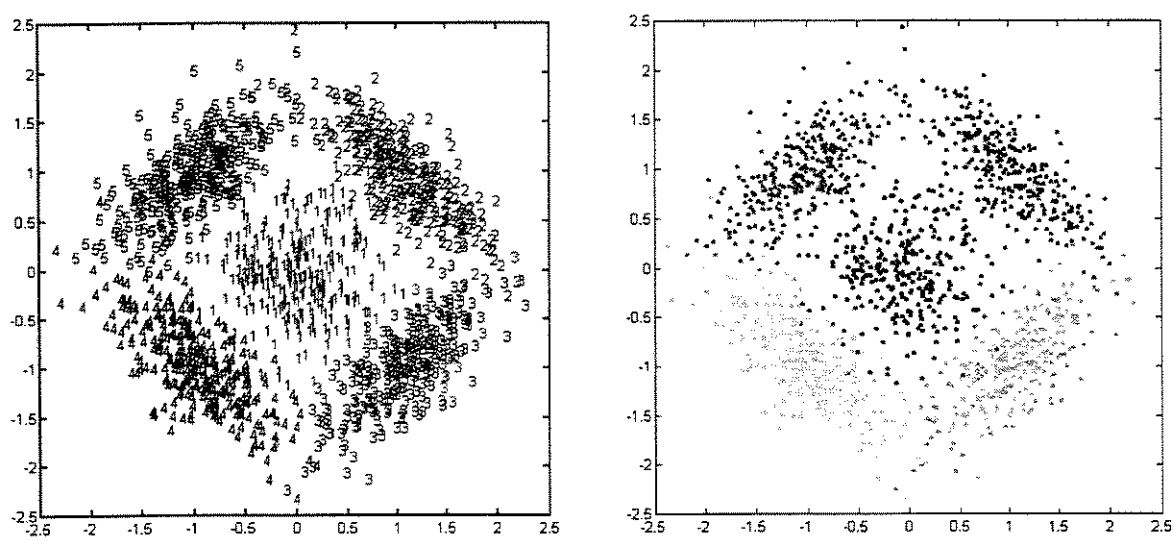


Figura 5.29 - Resultado do k-means para o conjunto de dados chainlink com k = 6.

5.5.2 Mistura de Gaussianas bivariadas

Um conjunto de dados para testes com redes neurais Gaussianas foi proposto por Hamad et al. (1996). Cinco classes foram geradas contendo cada uma 300 amostras. As cinco populações foram geradas a partir dos vetores de médias (0,0), (1,1), (1, -1), (-1, -1), (-1, 1). A matriz de covariâncias da primeira classe é diagonal, $\Sigma_1 = diag(0.2, 0.2)$ e as outras matrizes de covariâncias, também diagonais, foram obtidas usando $\Sigma = diag(0.05, 0.3)$ rotacionadas com ângulo $\pm \pi/4$. Os dados gerados são apresentados na figura 5.30. Vemos que há uma certa sobreposição nas classes, i.e., objetos de uma classe foram gerados em áreas de forte concentração de outra classe, o que sem dúvida torna o problema mais difícil.



(a) (b)

Figura 5.30: Ilustração do conjunto de dados: (a) representando objetos pela classe, e (b) representando objetos por diferentes cores.

Para usar o algoritmo *expectation-maximization* (*EM*), ver capítulo 2, inicialmente precisa-se determinar o número de classes. Alguns valores foram testados, numa faixa de $k = 2$ a $k = 8$, e o valor mais indicado, usando os critérios de informação (capítulo 4), foi $k = 5$. A figura 5.31 ilustra contornos eqüidistantes (usando a distância de Mahalanobis) dos centros, marcados com x, dos componentes da mistura de densidades de probabilidades.

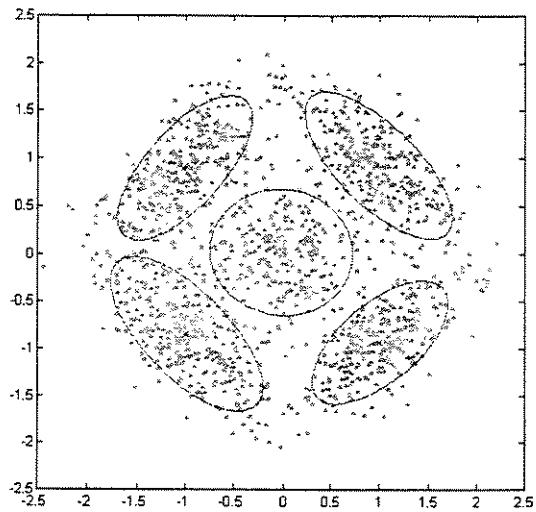


Figura 5.31: Contornos eqüidistantes dos centros das densidades componentes da mistura, determinadas pelo algoritmo EM.

Os vetores de médias obtidos foram, μ_1 = (-0.0203, -0.0218), μ_2 = (1.0396, 0.9724), μ_3 = (1.0376, -0.9789), μ_4 = (-0.9984, -0.9713), e μ_5 = (-1.0265, 0.9651). As matrizes de covariâncias estimadas foram

$$\Sigma_1 = \begin{bmatrix} 0.1695 & -0.0005 \\ -0.0005 & 0.1720 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 0.1908 & -0.1450 \\ -0.1450 & 0.1844 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 0.1793 & 0.1256 \\ 0.1256 & 0.1796 \end{bmatrix},$$

$$\Sigma_4 = \begin{bmatrix} 0.1822 & -0.1301 \\ -0.1301 & 0.1902 \end{bmatrix} \text{ e } \Sigma_5 = \begin{bmatrix} 0.1766 & 0.1136 \\ 0.1136 & 0.1616 \end{bmatrix}$$

e possuem valores próximos aos usados na geração dos dados. Os pesos dos componentes obtidos também ficaram muito próximos do ideal (0.20), π_1 = 0.1938, π_2 = 0.2021, π_3 = 0.2005, π_4 = 0.2026 e π_5 = 0.2010. Diferenças devem-se à aleatoriedade do processo de geração de dados.

Dois fatores contribuem para um bom desempenho do EM. Inicialmente, os dados são provenientes de um processo Gaussiano, e este é o modelo assumido para as misturas que estamos tratando. Desta forma, a geometria que o método impõe no espaço coincide com a geometria dos agrupamentos, resultando no sucesso da aplicação do método. O outro fator é que as médias estão relativamente longe entre si. Em geral o processo é bastante comprometido quando os centros dos componentes estão bastante próximos.

A função densidade de probabilidade da mistura é apresentada na figura 5.32, e a quantização do espaço é apresentada na figura 5.33. Note que os eixos estão com valores diferentes da figura 5.31. O ponto (0,0) da figura 5.31 está no centro da imagem apresentada na figura 5.33. Desta forma, as coordenadas da figura apresentada são as da imagem, porém estão diretamente relacionadas às coordenadas reais. Esta figura foi gerada discretizando o espaço de atributos, de $(x_{in}, y_{in}) = (-3, -3)$ a $(x_{fin}, y_{fin}) = (3, 3)$, usando um passo de discretização tanto em x quanto em y de 0.025, o que resultou em uma imagem de tamanho 241×241. Classificando cada ponto, ou pixel, de acordo com a maior estimativa da densidade de probabilidade de Bayes (equação 2.53) em relação aos componentes, obtemos as regiões das classes C_i, $i = 1, 2, \ldots, 5$. Estando o espaço de atributos rotulado, podemos, baseado na informação do modelo obtido, classificar, inclusive novas amostras não usadas na fase de treinamento. Porém, o objetivo inicial é a descoberta dos agrupamentos, o que foi obtido com sucesso. A figura 5.34 ilustra o resultado da aplicação do conjunto de dados ao modelo de agrupamentos obtido. Apesar da semelhança com a *confusion matrix*[9] dos sistemas de discriminação de padrões, neste caso os próprios dados foram usados para testar a eficiência da partição obtida. A interpretação é similar à da *confusion matrix.* A classe dos dados são representadas por linhas enquanto que as classes obtidas são representadas pelas colunas. Por exemplo, 13 padrões da classe 2 foram classificadas como pertencentes à classe 1. O ideal seria obter uma matriz diagonal, porém, isto não foi possível devido à sobreposição dos dados das classes, gerado a partir de um processo Gaussiano aleatório.

[9] Matriz utilizada em reconhecimento de padrões para indicar quantos padrões de cada classe foram atribuídos às classes. Caso seja diagonal, temos 100% de acertos. Para maiores detalhes consulte Duda *et al.* (1998) ou Ripley (1996).

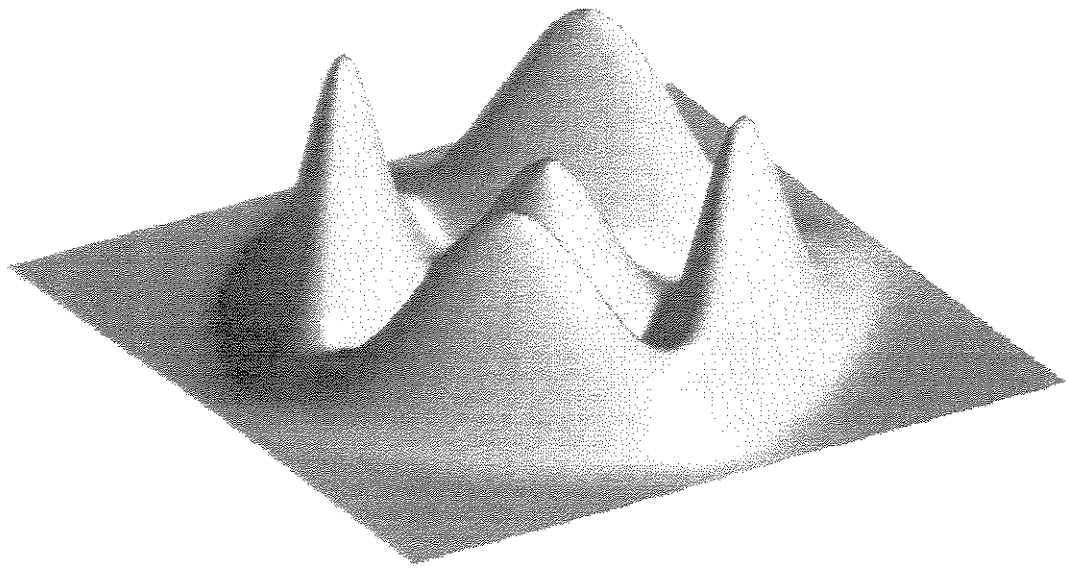


Figura 5.32: Função densidade de probabilidade da mistura

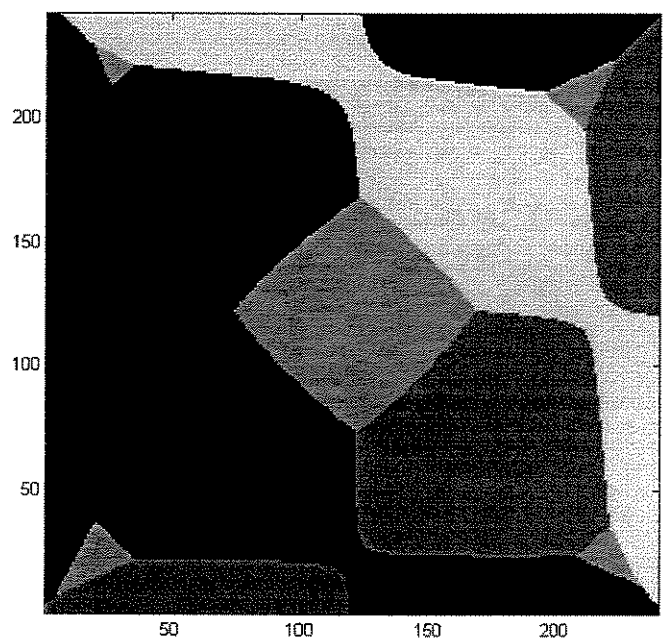


Figura 5.33: Quantização do espaço
Função densidade de probabilidade da mistura

294	1	0	2	3
13	273	5	5	4
0	3	293	3	1
2	1	5	292	0
1	1	2	0	296

Figura 5.34: Confusion matrix para teste da partição do espaço obtido pelo modelo de misturas de gaussianas usando o algoritmo EM.

O resultado da classificação é apresentado na figura 5.35, onde as cores (ou níveis de cinza) correspondem às classes descobertas pelo método.

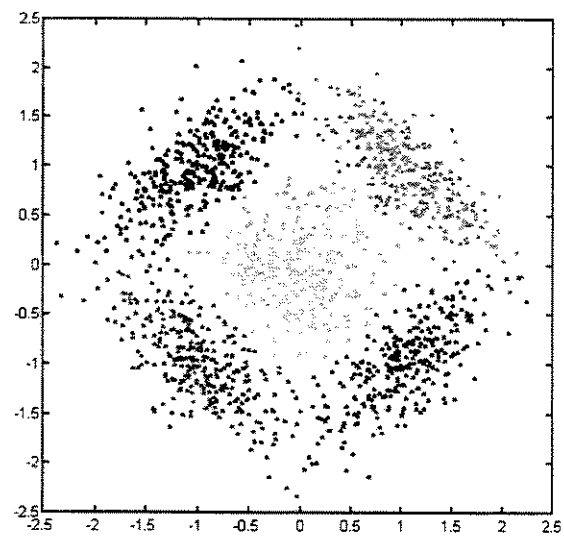


Figura 5.35: Resultado da classificação usando o modelo de misturas de Gaussianas. As classes dos objetos são representadas por cores diferentes.

Testamos o mesmo conjunto de dados com um SOM com dimensão 2 e tamanho do *grid* 12×12. O mapa foi inicializado de forma linear, e após 200 épocas de treinamento com o algoritmo de adaptação em lote, obtemos a configuração de neurônios como mostrada na figura 5.36. A função de vizinhança utilizada foi Gaussiana, onde o raio inicial usado foi 9, decrescendo até 1, de forma linear. Note que houve uma concentração de neurônios nas regiões de maior densidade de pontos, e que os dados foram escalonados ao intervalo [0, 1]. As figuras 5.37 e 5.38 ilustram o histograma de vencimentos dos neurônios. Note que neurônios em regiões de maior concentração de pontos possuem valores mais elevados neste histograma.

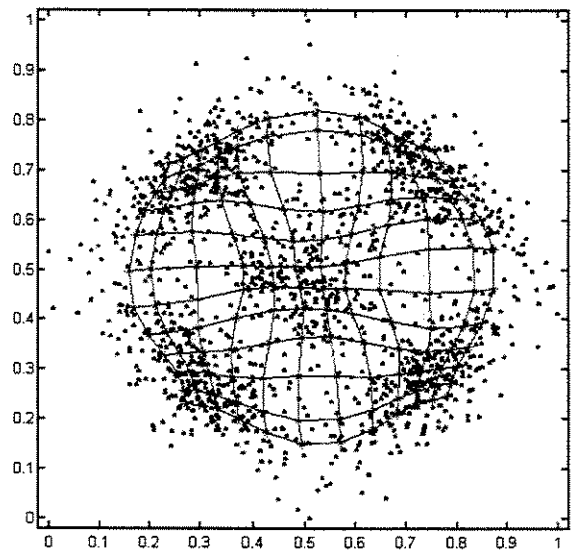


Figura 5.36: Grid de um som com dimensões 12×12 após 200 épocas de treinamento com o algoritmo batch.

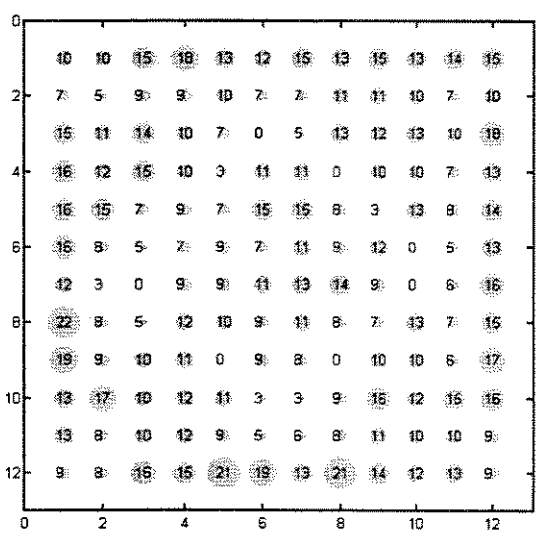


Figura 5.37: Histograma de vencimentos dos padrões pelos neurônios após o treinamento.

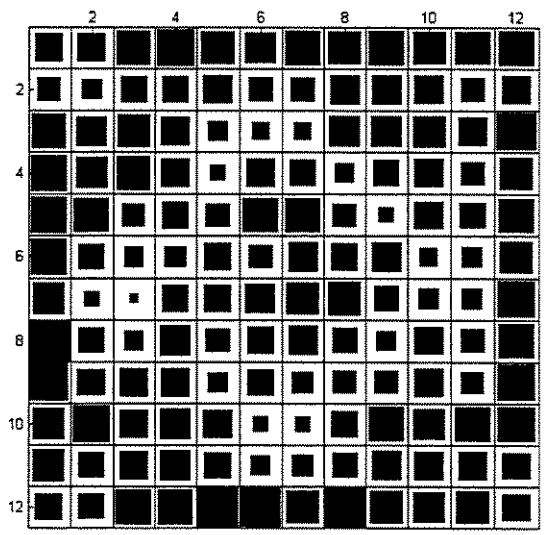


Figura 5.38: Histograma de vencimentos dos padrões pelos neurônios após o treinamento.

As figuras 5.39 e 5.40 ilustram a superfície de influências do mapa após treinamento, onde na figura 5.40 os dados estão representados por pontos. Como descrito no capítulo 3, cada neurônio possui uma influência isotrópica no espaço que é limitada apenas pelas influências dos neurônios vizinhos. A superfície de influências é um modo de ver o diagrama de Voronoï, onde plota-se a distância do centro dos neurônios no eixo z,

permitindo analisar, quando o problema tem dimensionalidade 2, o resultado da quantização vetorial. A figura 5.41 mostra a informação apresentada na figura 5.40 em 3-D.

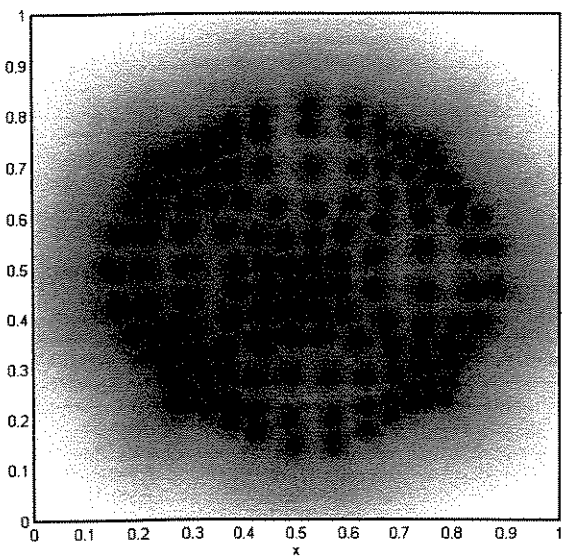


Figura 5.39: Superfície de influências do mapa após treinamento

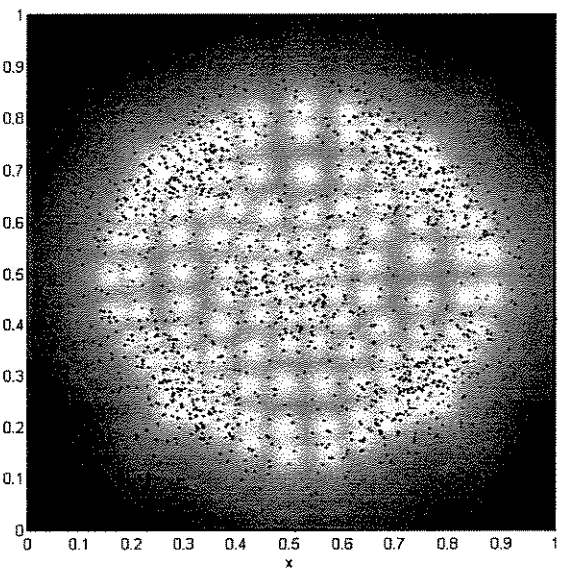


Figura 5.40: Superfície de influências do mapa com sobreposição dos dados

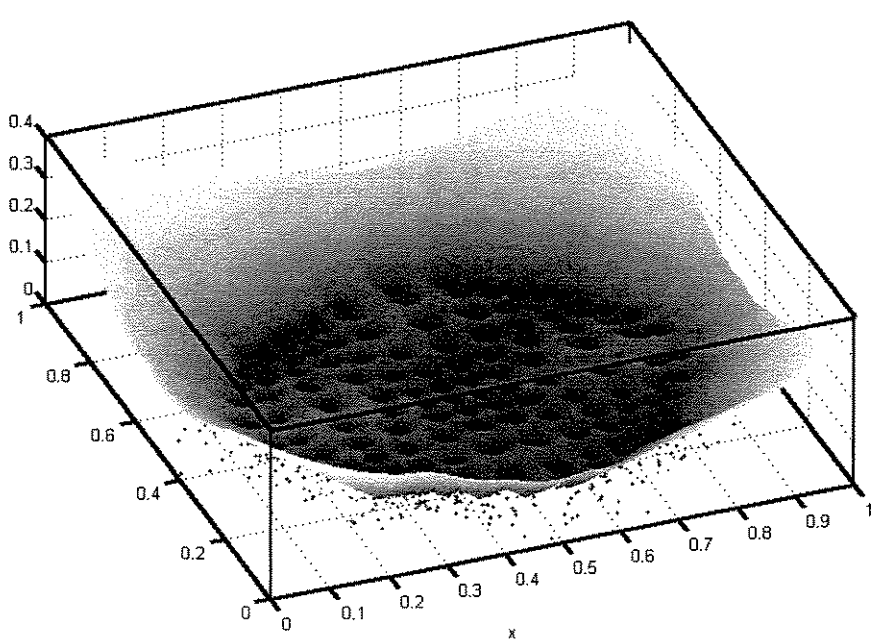


Figura 5.41: Superfície de influências do mapa em 3-D.

A *U-matrix* correspondente ao mapa treinado (configuração apresentada na figura 5.36) é apresentada nas figuras 5.42 e 5.43, esta última na forma 3-D. Note que a *U-matrix* apesar de visualmente detectarmos, por exemplo olhando a figura 5.42, que existem 5 agrupamentos, testes com vários métodos convencionais como threshold iterativo (Parker, 1997) não apresentaram bons resultados. Efetuando a análise como descrita pelo algoritmo SL-SOM, obtêm-se o gráfico de número de regiões conectadas *versus* limiar da *U-matrix* apresentado na figura 5.44.

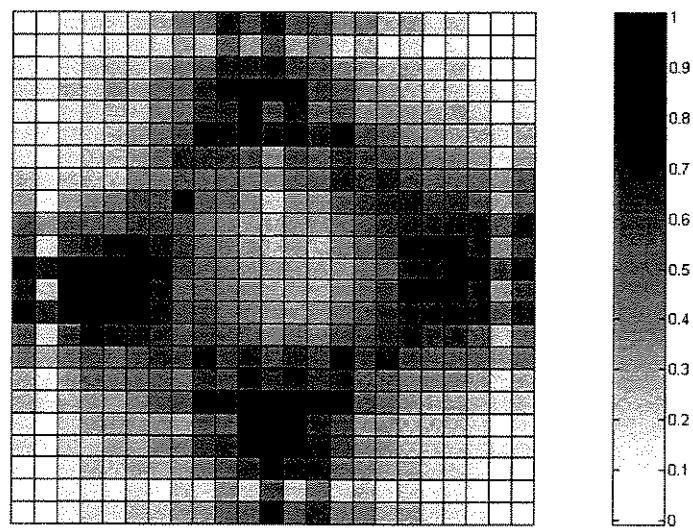


Figura 5.42: U-matrix do mapa apresentado na figura 5.36, em 2-D.

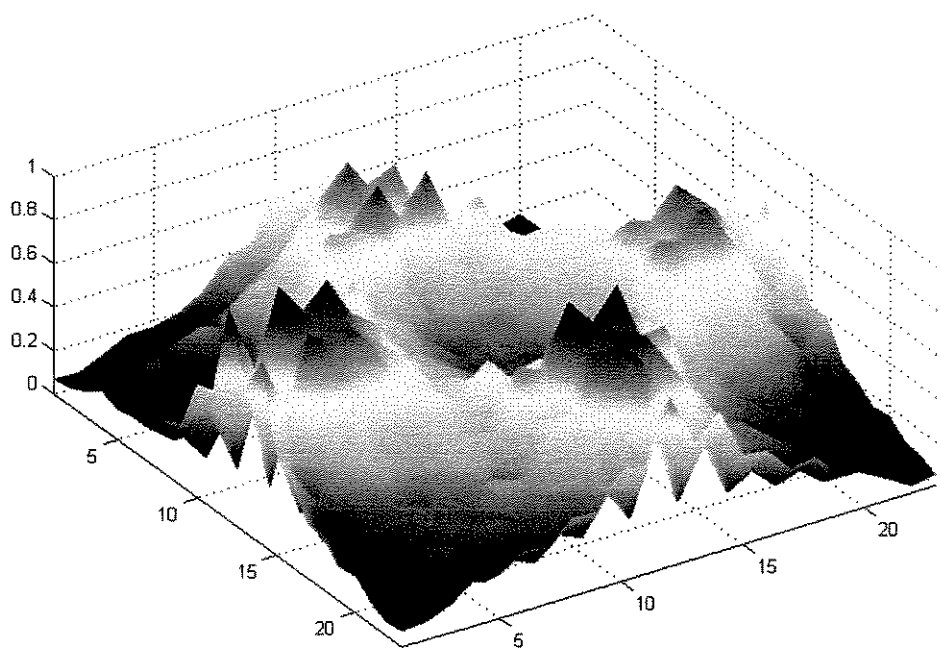


Figura 5.43: U-matrix do mapa apresentado na figura 5.36, em 3-D.

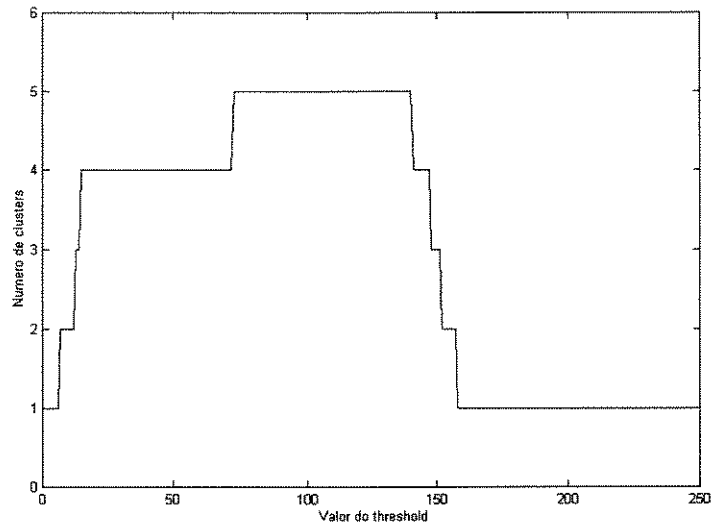


Figura 5.44: Gráfico de número de regiões conectadas versus limiar da U-matrix.

Nota-se na figura 5.44 que há duas grandes regiões de estabilidade, uma para 4 e outra para 5 agrupamentos, porém esta última venceu por uma diferença um pouco acima de 10% de níveis (valores de limiares) a mais. Uma vez escolhido o número de regiões, que implica diretamente no número de marcadores para a *watershed*, obtemos a imagem de marcadores como mostrada na figura 5.45. As linhas de *watershed* obtidas são mostradas na figura 5.46. A figura 5.47 ilustra a sobreposição das linhas de *watershed* sobre a *U-matrix* apresentada na figura 5.43.

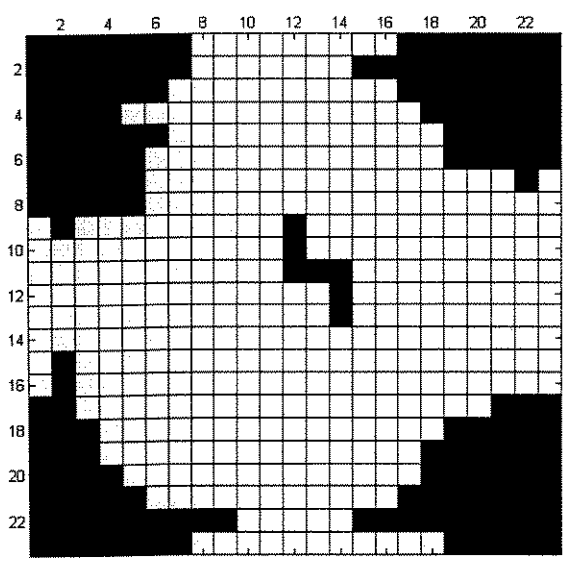


Figura 5.45: Marcadores escolhidos (em preto)

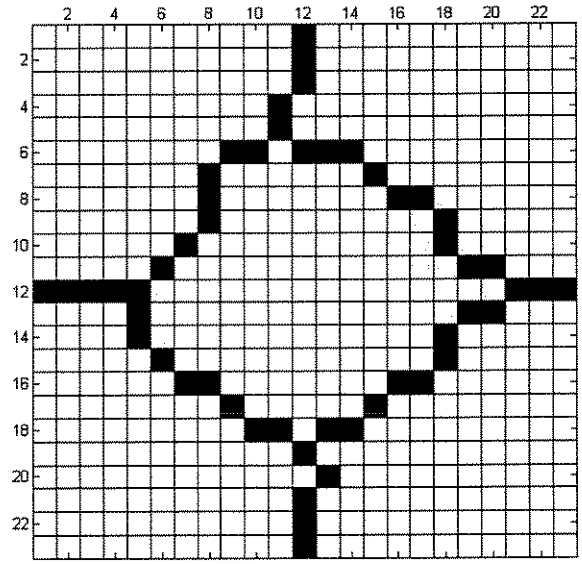


Figura 5.46: Linhas da watershed obtidas da U-matrix, usando os marcadores apresentados na figura 5.45.

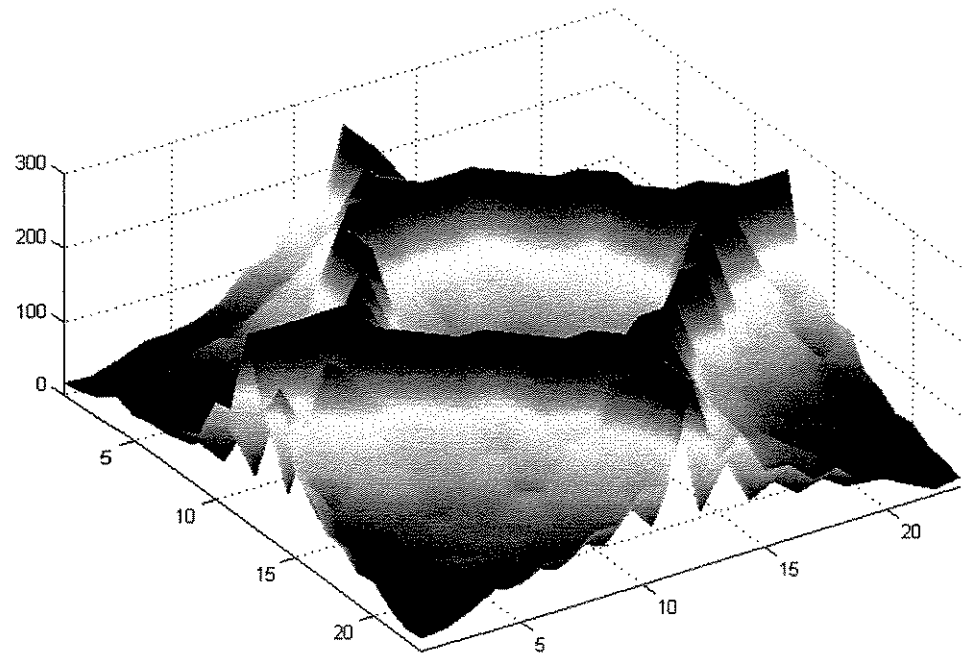


Figura 5.47: Sobreposição das linhas de watershed sobre a U-matrix apresentada na figura 5.43.

A figura 5.48 ilustra o resultado da aplicação do método rotulagem de regiões conectadas sob a segmentação obtida pelo *watershed* na *U-matrix* (figura 5.46), enquanto que a figura 5.49 mostra a cópia dos rótulos da *U-matrix* para os neurônios correspondentes da rede SOM. Note que, na figura 5.48, sete neurônios não estão rotulados, pois suas posições correspondentes na *U-matrix* fizeram parte do contorno entre as regiões.

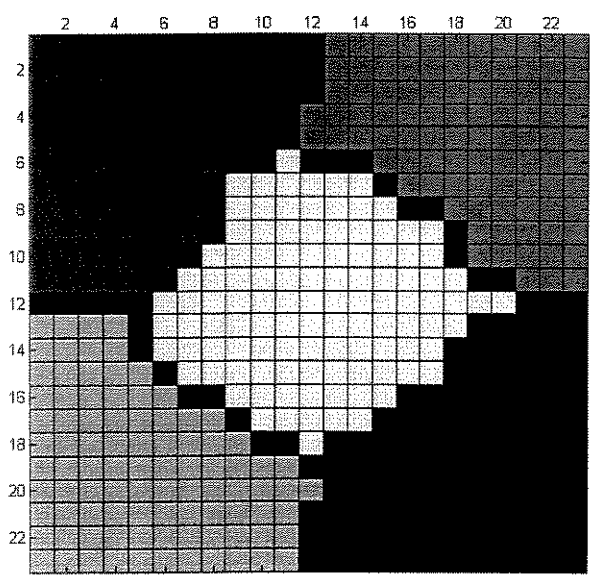


Figura 5.48: resultado da aplicação do método rotulagem de regiões conectadas na figura 5.46.

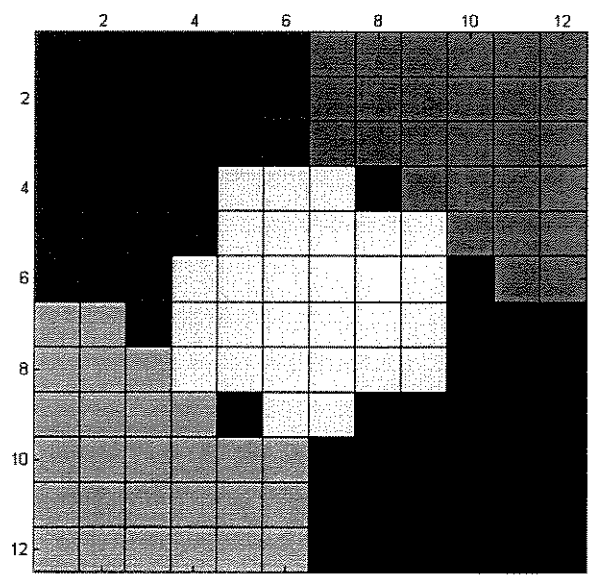


Figura 5.49: Mapa rotulado a partir dos códigos das regiões da U-matrix apresentada na figura 5.47.

A figura 5.50 mostra o mapa totalmente rotulado, onde foi aplicado o passo 6 do algoritmo SL-SOM, i.e., usando o rótulo do neurônio já rotulado e vizinho mais próximo, com distâncias calculadas no espaço de pesos, para rotular os sete neurônios remanescentes do processo de rotulagem da *U-matrix*. A figura 5.51 mostra a configuração de neurônios no espaço de pesos, onde cada neurônio é representado por um círculo, de tamanho proporcional ao número de vezes que mapeou padrões, e cores, que indicam a classe do neurônio. Novamente, note que alguns neurônios situados fora das regiões de maior densidade de pontos são pouco representativos no processo de agrupamentos, e mostraremos, mais adiante, que esta informação pode ser útil na geração de *U-matrizes*.

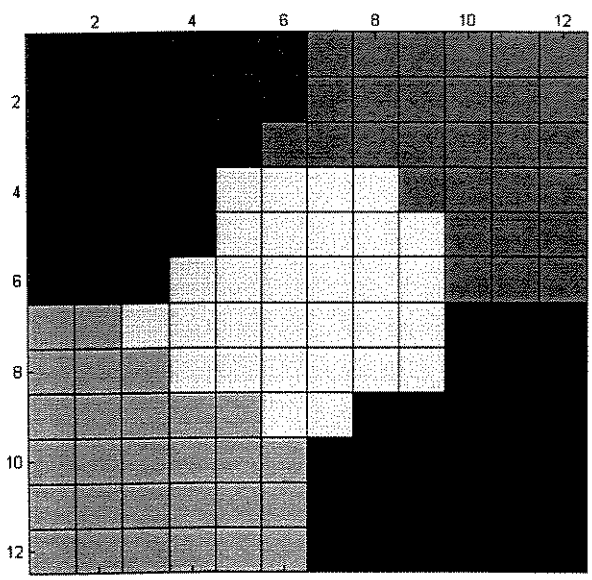


Figura 5.50: Mapa totalmente rotulado, usando o rótulo dos vizinhos mais próximos (distância calculada no espaço de pesos) nos neurônios que estavam não rotulados na figura 5.49.

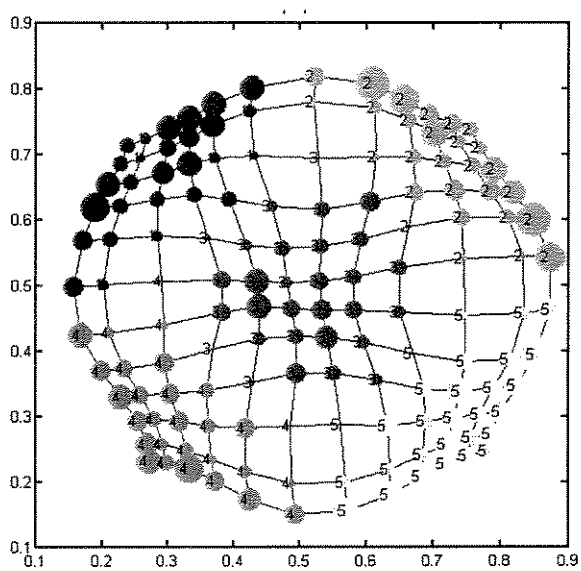


Figura 5.51: Configuração de neurônios no espaço de pesos, onde cores representam as classes dos neurônios e o tamanho do círculo representa o número de vezes que cada neurônio mapeou algum padrão.

A figura 5.52 mostra a *confusion matrix* obtida pelo método SL-SOM. Note que o resultado apresentado é muito próximo ao método de misturas de densidades de probabilidades usando o algoritmo EM (menos de 1%), i.e., 95.67% contra 96.53%, o que é excelente, pois não tivemos que supor que os dados proviam de distribuições Gaussianas, não foram usados métodos para determinar o número de densidades componentes (critérios de informação, baseados em estimativas da verosimilhança), nem houve necessidade de estimar os vários parâmetros das várias densidades componentes. O método das misturas

foi melhor devido à própria estrutura dos dados, que se encaixam perfeitamente no modelo geométrico dos protótipos obtidos. Para outros conjuntos de dados, como veremos adiante, e como é o caso no conjunto de dados apresentado na seção 5.5.1, o *chainlink*, o método das misturas falha por tentar impor uma estrutura Gaussiana na geometria complexa dos agrupamentos.

294	3	0	3	0
1	289	0	0	10
8	13	266	5	8
4	0	1	294	1
0	0	0	8	292

Figura 5.52: Confusion matrix obtida pelo método SL-SOM.

Quantizando o espaço de atributos, de forma similar ao que foi feito na figura 5.33 para o método das misturas, obtemos a figura 5.53, onde o espaço foi discretizado e rotulado de acordo com a proximidade aos protótipos dos agrupamentos de neurônios. Note, em relação à figura 5.33, que neste caso não há descontinuidades das classes no espaço de atributos, o que ocorre no caso das misturas. A descontinuidade é causada porque as densidades Gaussianas influenciam mais nas direções dos seus eixos principais. Nas direções onde há menor variabilidade a influência cessa mais rapidamente, e caminhando nesta direção a influência decresce a um valor que é inferior à influência de outra densidade, mudando-se a classe, naquele ponto. Isto porque cada ponto no espaço é classificado por uma regra semelhante à regra de Bayes, equação 2.53, onde calcula-se a probabilidade do ponto pertencer a cada um dos componentes e a classe atribuída é a classe da densidade componente que gerou a máxima probabilidade da amostra ter sido extraída. No caso do SL-SOM como as influências de todos os neurônios é igual, e isotrópica, i.e., a distância é Euclidiana, não haverá descontinuidades das classes no espaço de atributos, pois um ponto sempre será classificado na classe do neurônio mais próximo, não havendo hipótese de um neurônio mais distante ganhar a concorrência. Apesar da influência dos neurônios ser isotrópica, a influência de cada agrupamento de neurônios rotulados conjuntamente é arbitrária e complexa, e é descoberta automaticamente pelo método.

A figura 5.54 ilustra o processo de classificação dos objetos no SL-SOM. Um objeto é classificado pela classe do agrupamento de neurônios que possua o neurônio vencedor para ele. A figura 5.55 mostra o resultado final da classificação pelo SL-SOM, e que apresenta os dados sumarizados na figura 5.52. Caso a variabilidade das classes fosse um pouco menor, como veremos adiante, poderíamos ter obtido 100% de classificação correta. Os erros de classificação são devidos ao processo de geração de dados Gaussiano que ocasionou amostras de uma classe ocorrerem em áreas de concentração de outras classes.

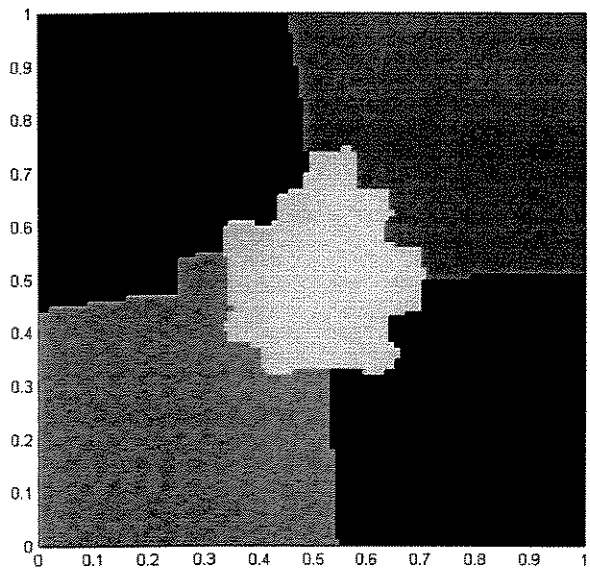


Figura 5.53: Quantização do espaço de atributos pelo SL-SOM

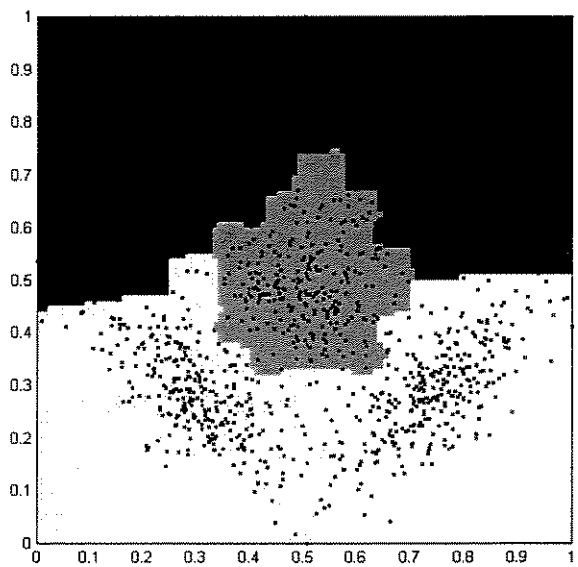


Figura 5.54: Quantização do espaço de atributos pelo SL-SOM e os dados.

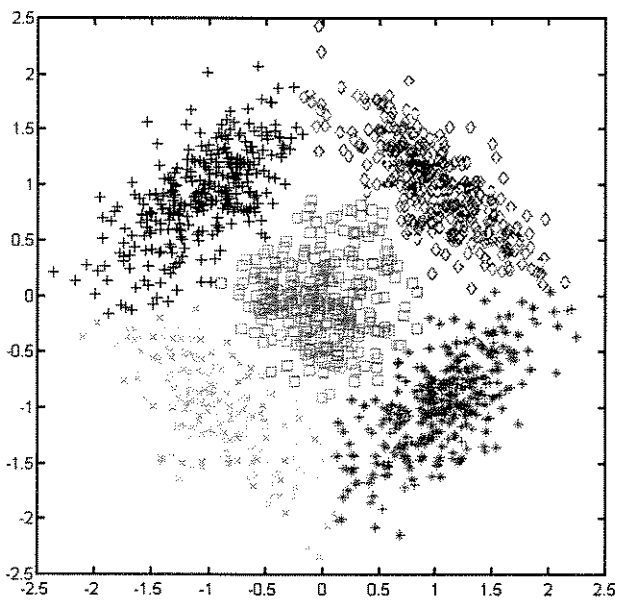


Figura 5.55: Resultado final da classificação dos objetos pelo SL-SOM

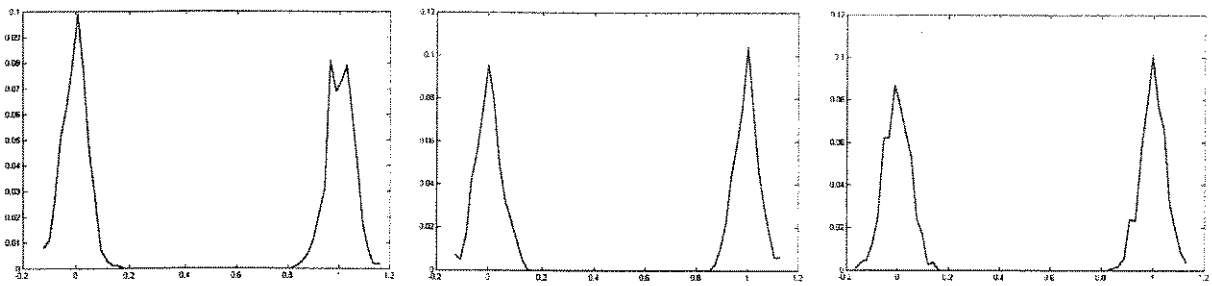
5.5.3 Mistura de Gaussianas no espaço $\Re^3$

O conjunto de dados apresentado a seguir foi usado em Costa e Netto (1998, 1999a) com dois propósitos fundamentais: analisar o efeito da sobreposição dos agrupamentos na degradação das bordas da *U-matrix* e implicações no processo do SL-SOM, e visualizar a deformação elástica da rede em um espaço de maior dimensão (3) do que o mapa (2).

Novamente o modelo de dados usado foi de mistura de Gaussianas. Três conjuntos de dados com 1000 objetos cada foram gerados contendo oito classes, na qual os vetores das médias correspondem a vértices de um cubo no espaço $\Re^3$, i.e., {(0,0,0), (0,0,1), ... (1,1,1)}. As matrizes de covariâncias usadas foram $\sigma^2_i \cdot I$, $i = 1, \ldots 3$, onde I é a matriz identidade. O parâmetro σ controla a variabilidade de cada conjunto de dados. Os valores de σ_i utilizados foram 0.05, 0.15 e 0.25, respectivamente para o primeiro, segundo e terceiro conjunto de dados.

A probabilidade de sobreposição das classes aumenta com o valor de σ. As figuras 5.56-a até 5.56-i mostram as densidades marginais das três variáveis para os três conjuntos de dados. Note que quando $\sigma = 0.05$, temos uma baixíssima probabilidade de sobreposição, pois o intervalo $\mu \pm 3\sigma$, correspondendo a 99.73% de chance dos números ocorrerem, corresponde a [-0.15, 0.15] e [0.85, 1.15], respectivamente para μ igual a 0 e a 1. Por outro lado, quando $\sigma = 0.25$, os intervalos $\mu \pm 3\sigma$ são [-0.75, 0.75] e [0.25, 1.75], ou seja, o limite superior do intervalo da variável (x, y, ou z) de uma classe quando esta variável tiver μ igual a 0 é superior ao limite inferior da classe que tenha para a mesma variável μ igual a 1, caracterizando a sobreposição mostrada nas figuras 5.56 (*g-i*).

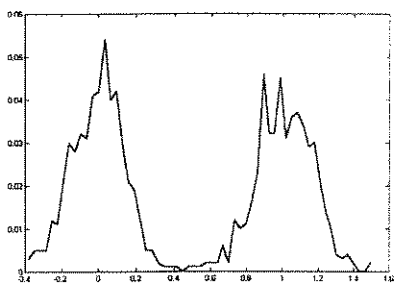
Nos experimentos efetuados com o SOM usamos mapas com tamanho 15×15. O mapa foi inicializado de forma linear, e o número de épocas nos três casos foi 1000, usando o algoritmo de adaptação em lote. A função de vizinhança utilizada foi Gaussiana, onde o raio inicial usado foi 12, decrescendo até 1, de forma linear com o número de épocas. A figura 5.57 ilustra a configuração do SOM após a inicialização linear para o conjunto de dados 1, i.e., $\sigma = 0.05$.



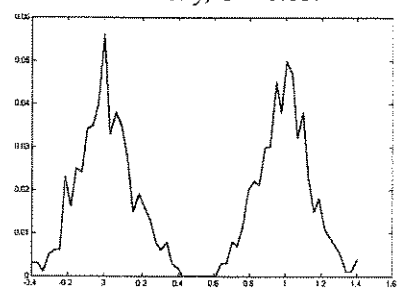
(a) Densidade marginal da variável x, $\sigma = 0.05$.

(b) Densidade marginal da variável y, $\sigma = 0.05$.

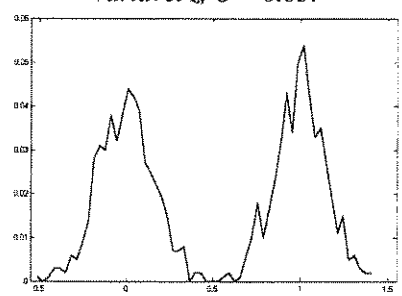
(c) Densidade marginal da variável z, $\sigma = 0.05$.



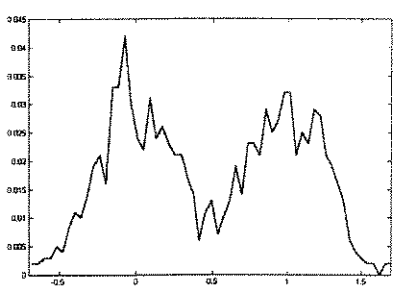
(d) Densidade marginal da variável x, $\sigma = 0.15$.



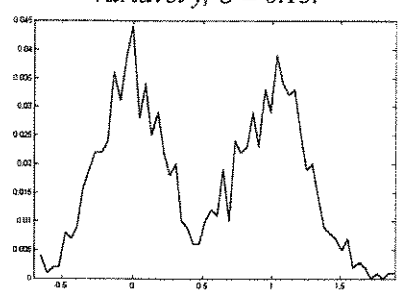
(e) Densidade marginal da variável y, $\sigma = 0.15$.



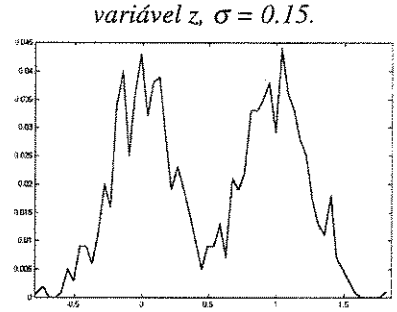
(f) Densidade marginal da variável z, $\sigma = 0.15$.



(g) Densidade marginal da variável x, $\sigma = 0.25$.



(h) Densidade marginal da variável y, $\sigma = 0.25$.



(i) Densidade marginal da variável z, $\sigma = 0.25$.

Figura 5.56: Densidades marginais (projeções) das variáveis x, y, e z dos três conjuntos de dados, $\sigma = 0.05$, 0.15 e 0.25..

A configuração dos neurônios obtida ao final do treinamento é apresentada nas figuras 5.58, 5.60 e 5.62, enquanto as U-matrizes correspondentes são mostradas nas figuras 5.59, 5.61 e 5.63, respectivamente para os casos onde $\sigma = 0.05$, 0.15 e 0.25. Note que a rede '*elástica*' do SOM tenta preencher o espaço tridimensional alocando mais neurônios nas regiões de maior densidade de pontos. Aumentando o valor de σ aumenta-se a degradação das bordas da *U-matrix*, dificultando o processo de detecção de agrupamentos pelo método SL-SOM. Bordas e vales ficam bem menos definidos e a situação extrema é quando a distribuição de dados provêm de uma densidade uniforme, onde ou há apenas um único agrupamento de todos os pontos, ou cada ponto constitui um agrupamento próprio.

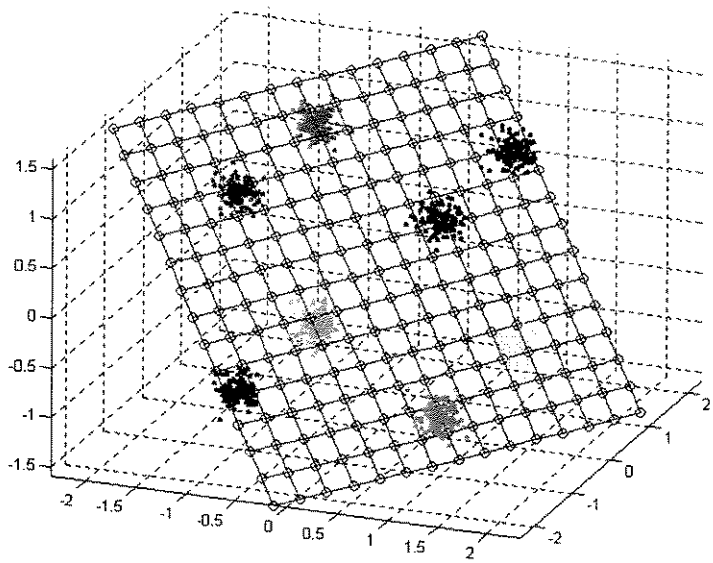


Figura 5.57: Configuração do SOM após a inicialização linear para o conjunto de dados 1, $\sigma = 0.05$.

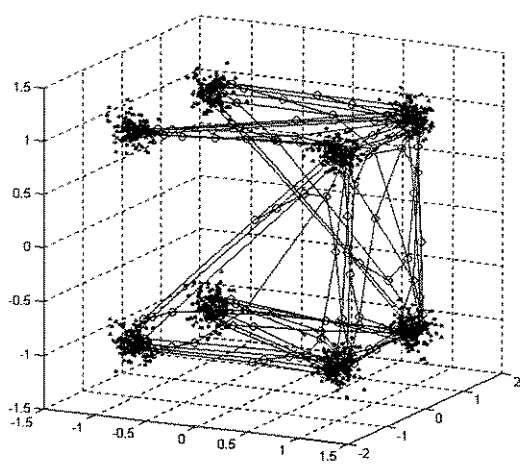


Figura 5.58: Configuração obtida para SOM 15x15 treinado com o conjunto de dados onde $\sigma = 0.05$.

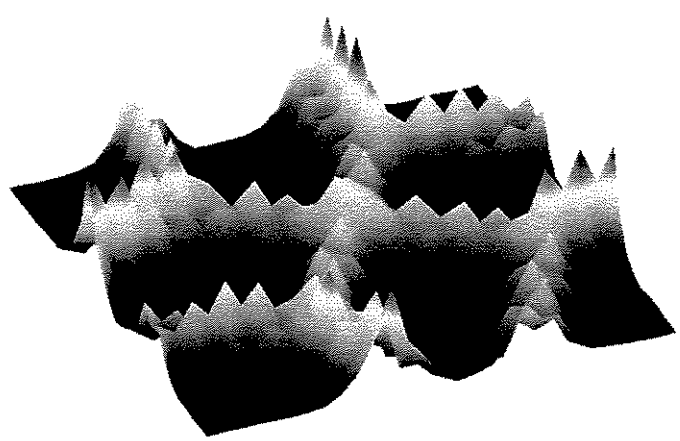


Figura 5.59: U-matrix relativa à configuração de neurônios apresentada na figura 5.58 ($\sigma = 0.05$).

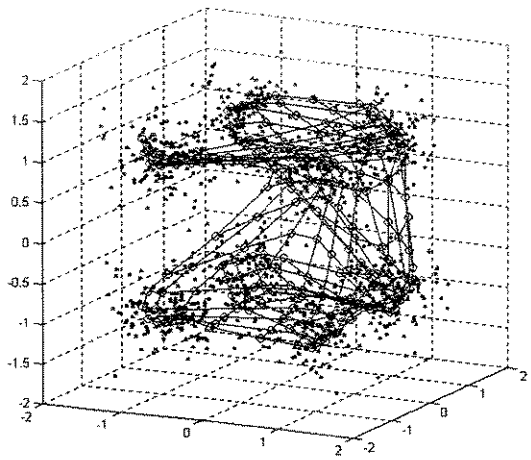


Figura 5.60: Configuração obtida para SOM 15x15 treinado com o conjunto de dados onde $\sigma = 0.15$.

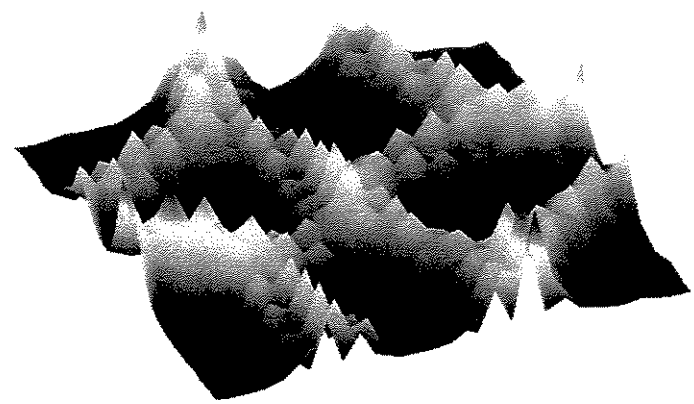


Figura 5.61: U-matrix relativa à configuração de neurônios apresentada na figura 5.60 ($\sigma = 0.15$).

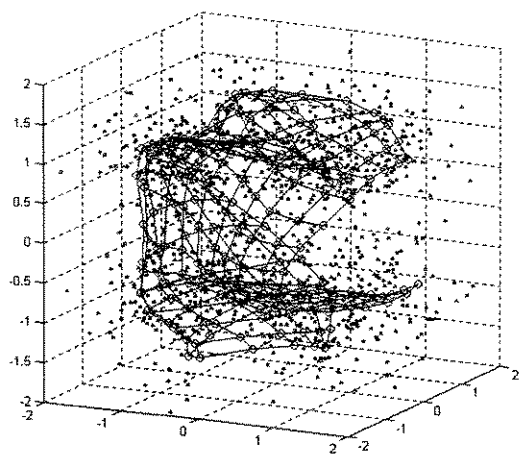


Figura 5.62: Configuração obtida para SOM 15x15 treinado com o conjunto de dados onde $\sigma = 0.25$.

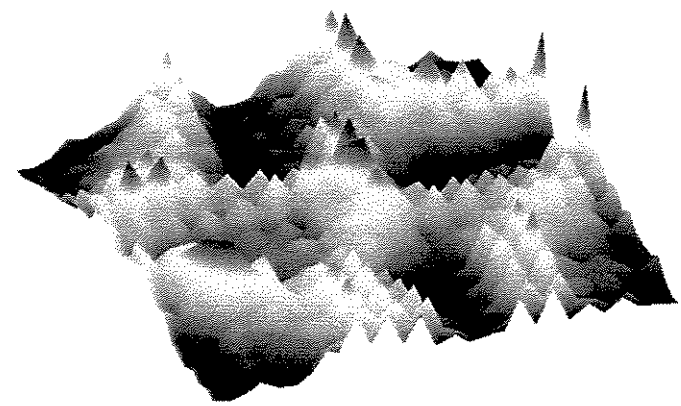


Figura 5.63: U-matrix relativa à configuração de neurônios apresentada na figura 5.62 ($\sigma = 0.25$).

O gráfico apresentado na figura 5.64 ilustra, para os três casos, o número de regiões conectadas para um valor crescente de limiar da *U-matrix*. O algoritmo apresentado na seção 5.4 detectou corretamente, para os três casos, o número de agrupamentos nos dados, 8. Isto é bem visível para o primeiro e segundo conjuntos de dados, que possuem grande estabilidade do número de regiões conectadas para um intervalo grande de valores de limiares da *U-matrix*. O terceiro conjunto de dados era esperado ter uma complexidade maior pelo fato da estrutura das classes geradas não ser tão bem definida quanto nos outros dois casos. Além do gráfico deslocar-se para a direita à medida que aumenta-se σ, nota-se também a instabilidade do número de regiões, ao redor do valor correto de agrupamentos no caso onde $\sigma = 0.25$. Neste conjunto de dados, a solução para N_{cr}^{k} igual a 8 venceu a solução N_{cr}^{k} igual a 9 por 16 níveis (intervalo 81-96) contra 12 (intervalo 58-69). Caso

deseje-se uma solução mais rigorosa, pode-se adotar um critério como margem de segurança, para aceitar soluções acima de determinada tolerância.

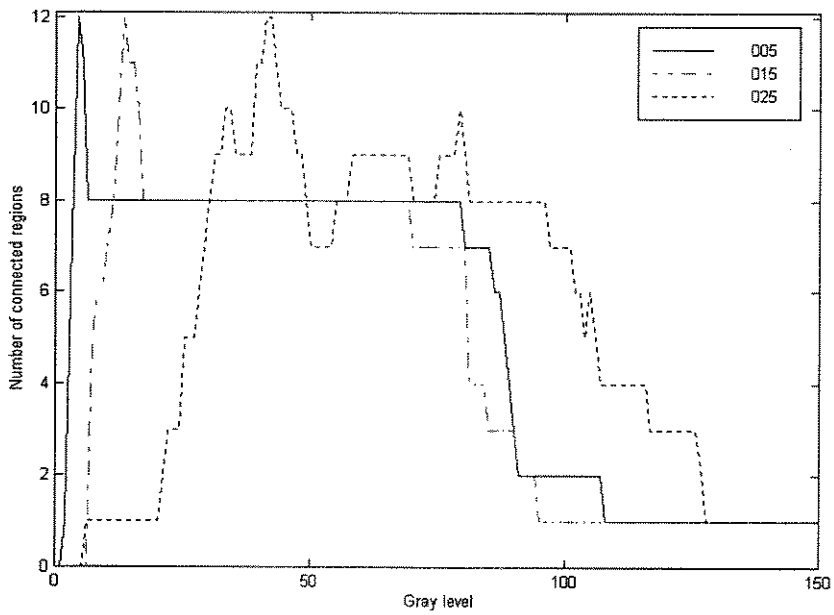


Figura 5.64: Gráficos do número de regiões conectadas versus limiar da U-matrix.

As imagens de marcadores para o *watershed* foram obtidas pela limiarização da U-matrix usando o nível inicial das regiões de estabilidade escolhidas, automaticamente, da figura 5.64. Os valores de limiares usados foram k = 6, 17 e 81, respectivamente para o primeiro, segundo e terceiro conjuntos de dados. As U-matrizes em níveis de cinza são apresentadas nas figuras 5.65, 5.67 e 5.69, enquanto que as figuras 5.66, 5.68 e 5.70 ilustram as partições do mapas, onde as regiões relativas a cada agrupamento estão rotuladas por níveis de cinza diferentes, e em preto estão apresentadas as linhas de *watershed* obtidas.

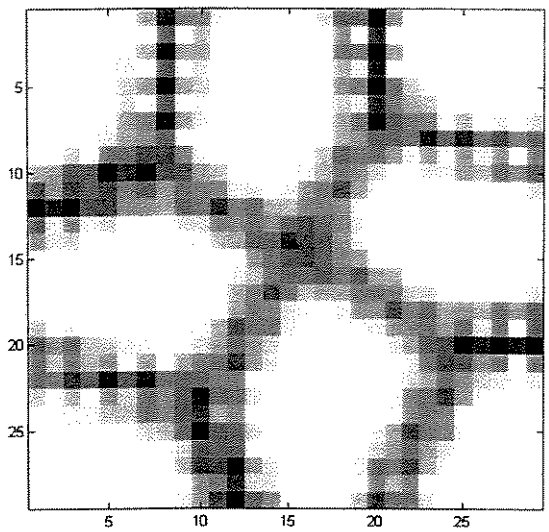


Figura 5.65: U-matrix equivalente a figura 5.59.

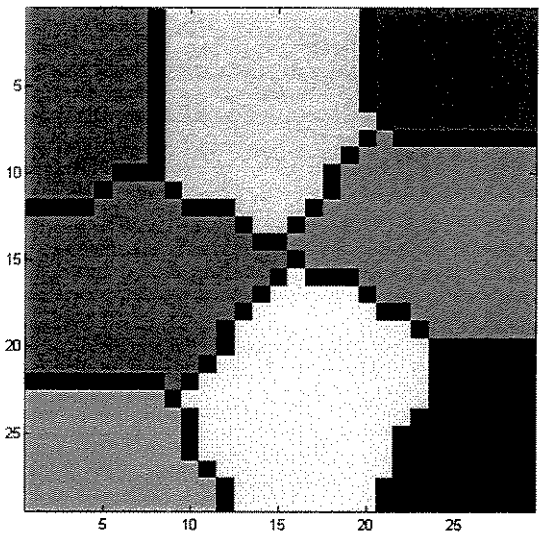


Fig. 5.66: U-matrix particionada e rotulada ($\sigma = 0.05$).

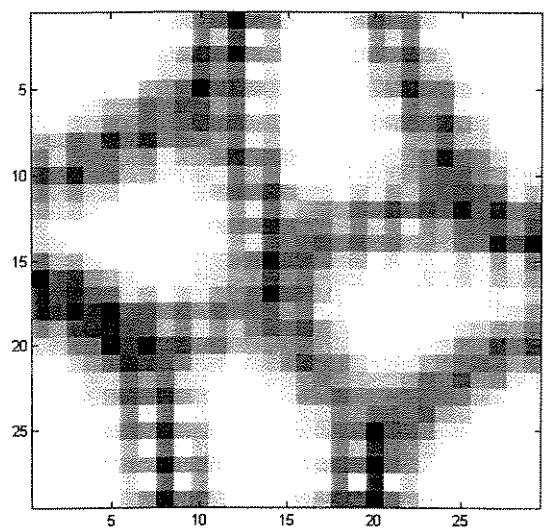


Figura 5.67: U-matrix equivalente a figura 5.61.

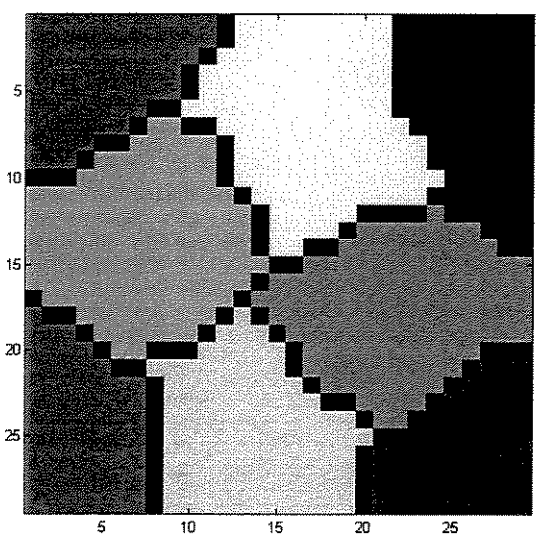


Fig. 5.68: U-matrix particionada e rotulada ($\sigma = 0.15$).

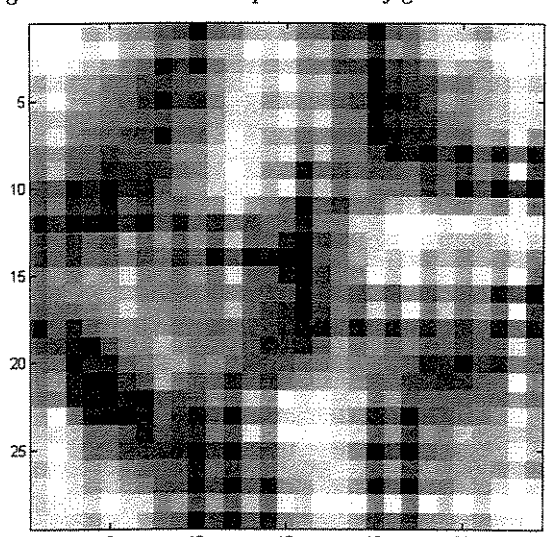


Figura 5.69: U-matrix equivalente a figura 5.63.

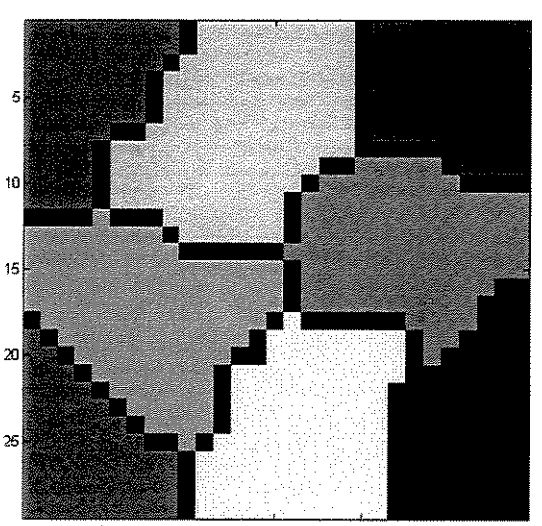


Fig. 5.70: U-matrix particionada e rotulada ($\sigma = 0.25$).

Testando a capacidade de identificar a estrutura dos agrupamentos detectados, classificando os padrões pelo rótulo da região que possui o neurônio mais próximo ao padrão obtivemos 100% de acertos no primeiro conjunto de dados, e 99.4% no segundo, i.e., apenas 6 objetos em 1000 foram alocados em classes não correspondentes ao processo que gerou os dados. Para o último conjunto de dados, obtivemos sucesso de alocação de padrões em 87.9%, e sua *confusion matrix* é apresentada na figura 5.71. A análise é direta, por exemplo, 16 amostras da classe 1 foram mapeadas na região da classe 4.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	***101***	0	1	16	0	0	7	0
2	0	***114***	5	4	2	0	0	0
3	6	4	***112***	0	2	0	0	1
4	4	2	0	***116***	0	3	0	0
5	0	4	0	0	***112***	6	0	3
6	0	0	0	6	1	***110***	6	2
7	1	0	0	1	0	0	***105***	18
8	1	0	4	0	11	0	0	***109***

Figura 5.71 - Confusion matrix para o conjunto de dados 3 ($\sigma = 0.25$).

Os resultados para estimação dos parâmetros das misturas de Gaussianas via EM foram superiores em termos de classificação a partir da estrutura encontrada: como no SL-SOM, no primeiro conjunto de dados obtivemos 100% de sucesso. No segundo caso apenas um objeto foi classificado erroneamente, enquanto que no terceiro 83 objetos foram classificados em regiões distintas da região original, resultando em 91.7% de acerto. Novamente há de se enfatizar que o modelo dos dados encaixa-se ao modelo estatístico, i.e., foram inclusive gerados a partir dele, e era de se esperar uma taxa de acertos maior. Porém, nos três casos, simulações foram efetuadas para vários valores de k para se determinar o número adequado de componentes da mistura, além do custo computacional de estimação das vários parâmetros das misturas. O SL-SOM conseguiu gerar o modelo de representação de protótipos distribuídos de forma adequada, e sem supor formas paramétricas das distribuições dos dados, obtendo resultados compatíveis ao EM.

A deterioração das bordas da U-matrix à medida que σ aumenta decorre do fato de que os neurônios da rede '*elástica*' não se separam tanto como no caso onde σ é pequeno, pois há maior representatividade de objetos na área central do hipercubo. Aumentando-se ainda mais σ é possível que o SL-SOM não detecte as oito classes como ocorreu para os três casos apresentados. Porém, problemas irão ocorrer com os outros métodos também. No

caso do *k-means*, em alguns experimentos, mesmo para o primeiro conjunto de dados e fazendo $k = 8$, ocorreram problemas como duas classes estarem representadas por um protótipo entre elas, enquanto que uma das outras classes remanescentes foi dividida entre dois protótipos, resultando em uma grande falha do método em descobrir a estrutura dos dados. Uma das causas é a possibilidade do *k-means* ficar preso em mínimos locais durante a otimização pela busca dos melhores protótipos. Em geral, com o *k-means*, sugere-se que sejam feitas várias simulações, partindo-se de condições iniciais diferentes, e a coincidência de vários resultados sugere a solução para o problema. No caso do SOM isto é menos crítico pois o método é menos sensível às condições iniciais, e a ordem de apresentação dos dados, caso estejamos usando o algoritmo de adaptação de pesos em lote. Além disso, não supomos nenhum valor de número de agrupamentos para o SL-SOM, o que é necessário nos métodos convencionais.

5.6 Uma breve interpretação do funcionamento do SL-SOM

Apesar de ser matematicamente complexa, iremos, nesta seção, abordar brevemente (e conceitualmente) o funcionamento do algoritmo SL-SOM. Inicialmente, assume-se que o treinamento tenha ocorrido com sucesso, i.e., a grade elástica de neurônios tenha se expandido de forma a representar o espaço de atributos topologicamente, e concentrando neurônios em regiões de maior densidade de objetos.

O conjunto de dados $\boldsymbol{X}$ é representado por um conjunto de neurônios $\boldsymbol{m}_i$, $i = 1, \ldots, N$, onde cada $\boldsymbol{m}_i$ é um vetor no espaço p-dimensional de atributos e N é o número total de neurônios da rede. O algoritmo SL-SOM agrupa os neurônios que estão próximos no espaço de atributos (distância calculada usando os pesos sinápticos), e ao mesmo tempo que são adjacentes, ou vizinhos, na grade da rede SOM. Isto é devido à *U-matrix* que pode ser vista como um dendograma do um método hierárquico LS (ligações simples) onde apenas as distâncias entre neurônios vizinhos na grade são calculadas na matriz de dissimilaridades, D_{ij} (ver capítulo 2). Assim, segmentar a *U-matrix* é equivalente a cortar o dendograma do método ligações completas com conectividade restrita (*contiguity constrained*) à vizinhança imposta pela topologia da rede, método que denominaremos LS-CRV. Os elementos da matriz de dissimilaridades D_{ij} correspondentes a neurônios não vizinhos na grade podem ser fixados em um valor bastante elevado, não interferindo no processo de fusão dos agrupamentos.

A desvantagem em usar LS-CRV a favor do SL-SOM é que ainda temos os mesmos problemas dos métodos hierárquicos. Por exemplo, uma rede com $N = 200$ neurônios

produz uma matriz de dissimilaridades com tamanho 200×200, que a cada fusão de agrupamentos reduz em uma linha e uma coluna até possuir apenas um agrupamento. Cada vez que dois agrupamentos são unidos deve-se usar uma ultramétrica para calcular as distâncias do novo agrupamento a todos os outros remanescentes na matriz de dissimilaridades. Além disto, a visualização da árvore apresentada em um dendograma não é tão eficaz quanto à apresentada na *U-matrix*, além do que estamos interessados em sistemas que possam detectar agrupamentos dentro de agrupamentos (como será mostrado no capítulo 6). A *U-matrix* provê um meio bastante interessante do usuário interfacear as informações de um banco de dados. Uma das extensões deste trabalho será a implementação de uma interface onde o usuário desloca o cursor do mouse na *U-matrix* (segmentada ou não) e janelas dinâmicas de informações resumidas do banco de dados que estão sendo mapeadas no neurônio da posição atual do cursor são apresentadas. O uso desta interface juntamente com o SL-SOM e sua extensão, o *Tree-Structured* SL-SOM habilitarão, em um futuro próximo, novas formas de interfacear e de pesquisar informações em bancos de dados, inclusive para buscas na *internet*.

5.7 Gerando bordas mais definidas na *U-matrix*

Sendo um método baseado no modelo de agrupamento vizinhos mais próximos restrito a vizinhança da rede, como descrito na seção anterior, temos, de uma certa forma, a mesma flexibilidade de detectar agrupamentos alongados ou de formas complexas (ver por exemplo a seção 5.5.1) que o LS possui, e que outros métodos falham, como é o caso dos métodos que se baseiam na busca do centróide mais próximo, por exemplo o *k-means* e as misturas de densidades Gaussianas. Porém, é conhecido o efeito cadeia que o método LS possui, i.e., caso existam pontos (mesmo que poucos) entre duas regiões de elevada concentração de pontos que caracterizem agrupamentos distintos, o método pode ligar os dois agrupamentos distintos pelo trajetória no espaço destes pontos de ligação (veja a figura 5.72). Ao invés de detectar dois agrupamentos, o método LS falha quando há tal tipo de configuração nos pontos. O efeito cadeia, ou encadeamento (*chaining*) pode ser visto como o problema '*amigos de amigos*'. Por exemplo, o objeto 1 é similar ao objeto 2 que é similar ao objeto 3, e assim por diante, o objeto *n-1* é similar ao objeto *n*. O efeito cadeia pode fazer com que todos estejam em um mesmo agrupamento, apesar do objeto 1 poder ser bastante diferente do objeto *n*.

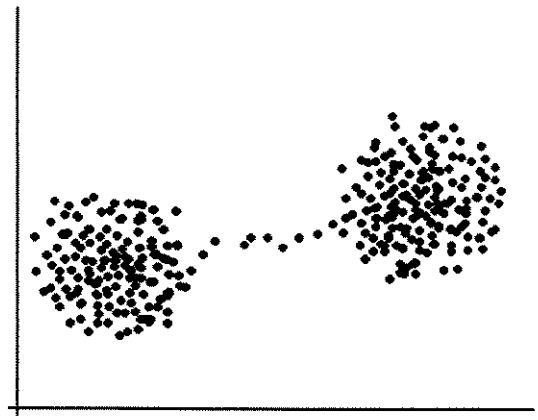


Figura 5.72: Pontos ligando dois agrupamentos distintos - o problema do efeito cadeia do método LS.

O problema similar ocorre no SOM. Olhando as figuras 3.20 e 3.22, nota-se a existência de neurônios de ligação, que inclusive vencem para poucos ou nenhum padrão, pois estão em regiões de baixa densidade de pontos. A *U-matrix* convencional, como apresentada na seção 5.1 usa a configuração estática dos neurônios para cálculo das distâncias, não importando-se com a informação de quantização dos padrões que cada neurônio está efetuando.

Uma maneira de melhorar as bordas da *U-matrix* é buscar eliminar o efeito destes neurônios de ligação, veja a figura 3.22 (neurônios representados por +), que apresentam um baixo número de vencimentos de padrões. Definimos um grau de ativação na seção 3.6 como um valor mínimo, φ, que os neurônios deveriam possuir para serem considerados para análises do mapa. Neurônios inativos podem ser definidos como neurônios cujo número de vencimentos seja inferior a φ, i.e., se $H(i, j) \leq \varphi$. Porém, eliminar neurônios inativos de um mapa bidimensional causará um buraco na rede e irá comprometer o cálculo da U-matrix, pelo menos nos moldes descritos na seção 5.1. Uma solução encontrada para eliminar, ou atenuar, o efeito de cadeia dos neurônios inativos é empurrá-los na direção dos neurônios ativos mais próximos. Isto pode ser feito de forma relativamente simples, considerando o histograma de vencimentos dos neurônios H, e copiando o vetor do neurônio ativo mais próximo ao neurônio inativo.

Seja N o número de neurônios na rede, e H o histograma de vencimentos dos neurônios, i.e., quantos padrões cada neurônio quantizou. Os passos da geração de uma nova configuração de neurônios, para um nível mínimo de ativação φ pode ser brevemente descrito como:

(Algoritmo: eliminação do efeito de cadeia dos neurônios inativos - EECNI)

Para $i = 1, 2, \ldots, N$

Se $H(i) \leq \varphi$

$\boldsymbol{m}_i \leftarrow \boldsymbol{m}_j$, onde $\| \boldsymbol{m}_i - \boldsymbol{m}_j \| < \| \boldsymbol{m}_i - \boldsymbol{m}_k \|$, $k = 1, 2, \ldots, N$, $k \neq j$, k $\neq i$, e $H(j) > \varphi$.

Fim-Se

Fim-Para

Note que o vetor no espaço p-dimensional $\boldsymbol{m}_j$ é copiado para o neurônio inativo $\boldsymbol{m}_i$. A busca do neurônio j inicialmente se restringe apenas aos neurônios adjacentes ao neurônio i, ou seja, em um raio de tamanho 1 a partir do neurônio i. Caso não seja encontrado neste raio nenhum neurônio ativo, para as condições especificadas, o raio é incrementado em uma unidade e a busca deve ser reiniciada, indo até quando houver um neurônio ativo que possa copiar seu vetor de pesos para o neurônio inativo.

A figura 5.73 ilustra o resultado do treinamento de um SOM bidimensional com tamanho 12×12 após 500 iterações do algoritmo em lote. O conjunto de dados usado (denominado anular) é formado por 1500 pontos distribuídos uniformemente em uma região circular do $\Re^2$. Note a existência de neurônios de ligação em regiões onde a densidade de pontos é zero (no centro). Usando o algoritmo EECNI obtêm-se, para $\varphi = 0$, a configuração de neurônios como mostrada na figura 5.74. É interessante notar que o SL-SOM detectou a presença de apenas um agrupamento, em ambos os casos, e em particular a diferença não foi muito significativa na *U-matrix*, porque as distâncias entre os neurônios que estão em regiões com densidade de pontos elevada é bem inferior que as distâncias aos neurônios situados no centro da região anular. A U-matrix original (derivada da configuração de neurônios apresentada na figura 5.73) é apresentada na figura 5.75.

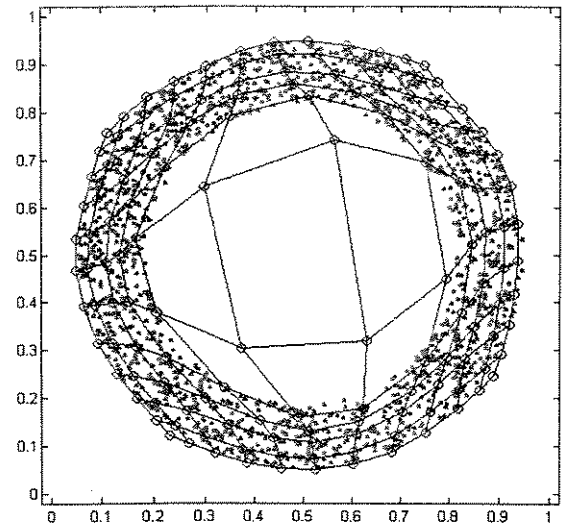


Figura 5.73: SOM 12×12 após 500 iterações (batch)

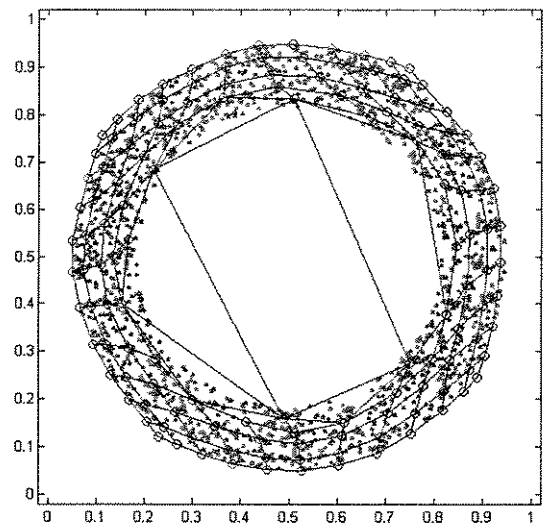


Figura 5.74: Aplicando o algoritmo EECNI na configuração apresentada na figura 5.73 com $\varphi = 0$.

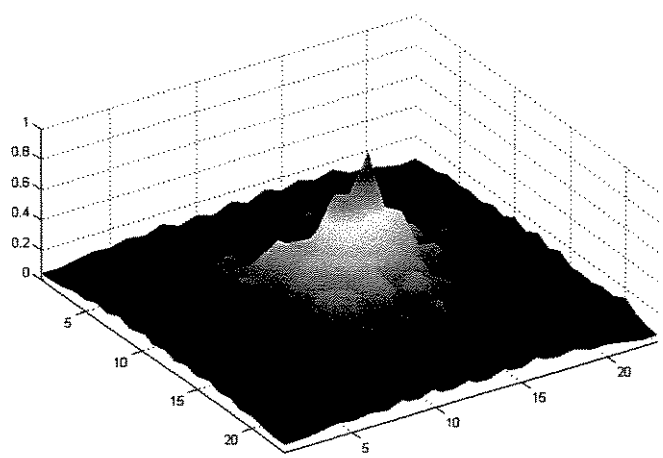


Figura 5.75: U-matrix correspondente ao SOM apresentado na figura 5.73 (em 3-D)

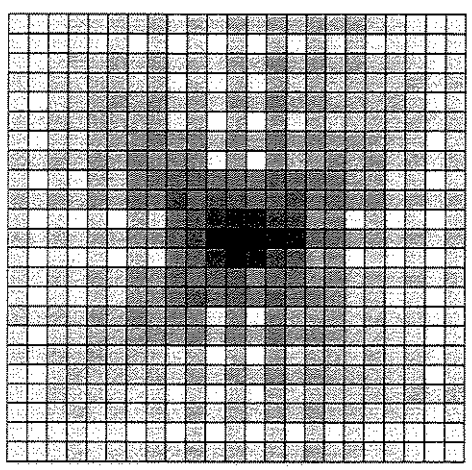


Figura 5.76: U-matrix correspondente ao SOM apresentado na figura 5.73 (em 2-D)

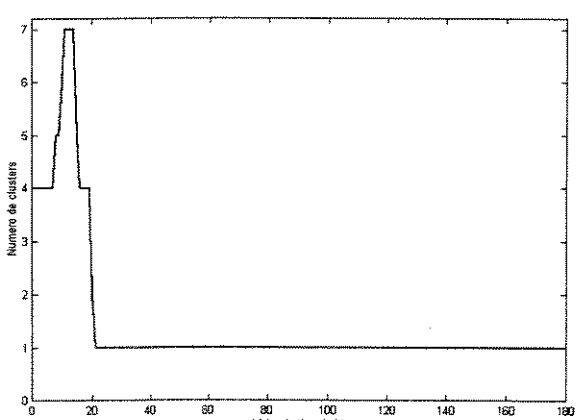


Figura 5.77: N_{cr}^{k} para o SOM apresentado na figura 5.73

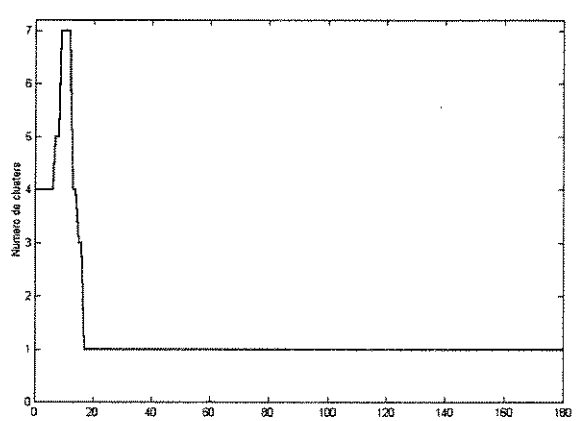


Figura 5.78: N_{cr}^{k} para o SOM apresentado Na figura 5.74

Note a pequena diferença entre as figuras 5.77 e 5.78, que representam o número de agrupamentos *versus* o limiar da *U-matrix*. No nível 17 (do intervalo 0 a 255), na figura 5.78, apenas uma região é estável, o que implica haver apenas uma classe nos dados. Já aplicando métodos como misturas de Gaussianas neste exemplo há problemas já no início onde métodos de determinação do número de componentes não funcionam bem. Por exemplo, simulando os critérios de informação (ver capítulo 4), para diversos valores de k, por exemplo $k = 1, 2, \ldots, K_{max}$, todos os critérios apontaram o valor K_{max} como o número de componentes. Foram testados diversos valores de K_{max} e os resultados se repetiram.

Em outro experimento, fazendo $\varphi = 3$ para o SOM apresentado na figura 3.20, cuja configuração é reapresentada na figura 5.79, obtemos, eliminando os neurônios que ganharam 3 ou menos vezes, através do algoritmo EECNI, a configuração de neurônios mostrada na figura 5.80. Note que o efeito resultante é como se os neurônios em posições de ligação entre as classes fossem eliminados, lembrando a figura 3.29.

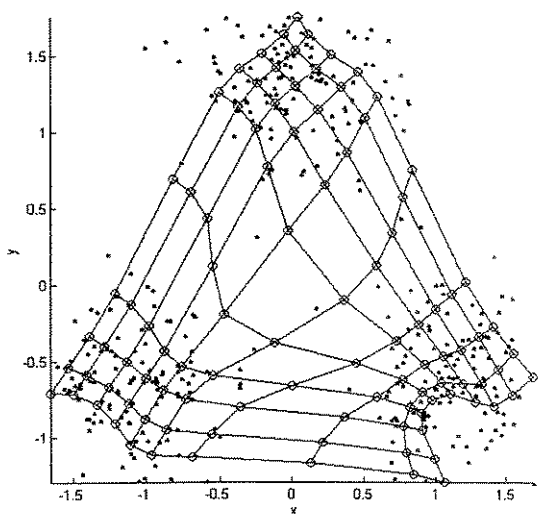


Figura 5.79: Configuração dos neurônios e os padrões, similar à figura 3.20.

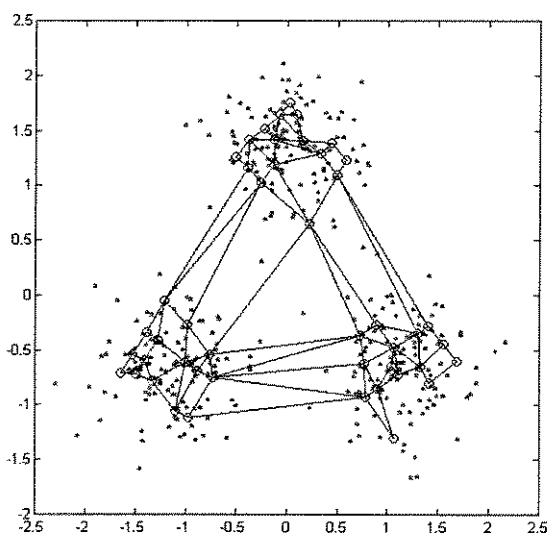


Figura 5.80: Configuração dos neurônios da rede apresentada na figura 5.79 após aplicação do EECNI com $\varphi = 3$.

Note a diferença nas *U-matrizes*, mostradas nas figuras 5.81 e 5.83 para o caso convencional e 5.82 e 5.84 quando aplicado EECNI *com* $\varphi = 3$. A borda está mais afinada na figura 5.82 do que na figura 5.81. O gráfico do número de regiões conectadas *versus* limiar da *U-matrix* para a imagem apresentada na figura 5.82 é mostrado na figura 5.85. Novamente percebe-se um platô significativo para 3 agrupamentos.

Porém, há de se notar que a intenção aqui não é desprezar a configuração original de neurônios, obtida via treinamento, e sim, uma configuração auxiliar na descoberta do número de agrupamentos corretos, evitando o efeito de cadeia, para então segmentar a *U-matrix* do mapa original, mesmo com os neurônios inativos, pois estaremos neste trabalho respeitando a quantização do espaço p-dimensional efetuado pelo SOM.

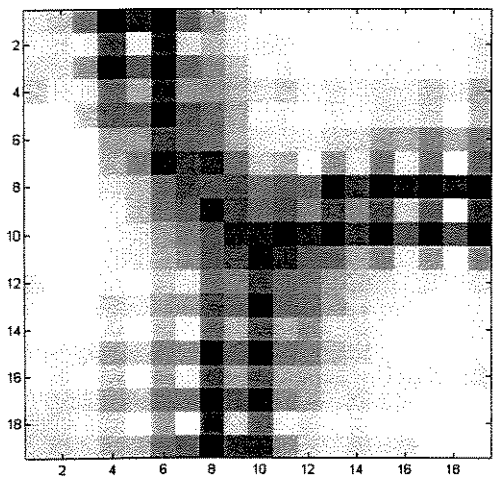


Figura 5.81: U-matrix (2D) para o SOM apresentado na figura 5.79.

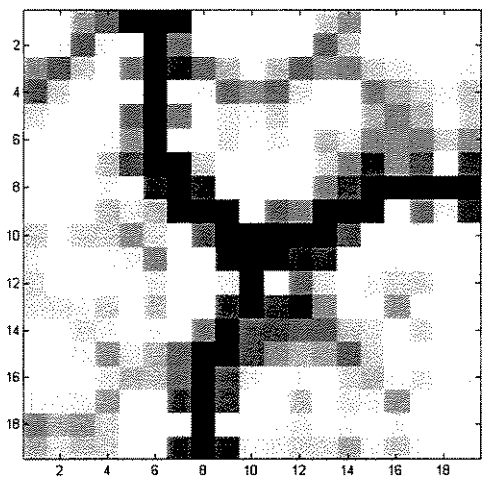


Figura 5.82: U-matrix (2D) para o SOM apresentado na figura 5.80.

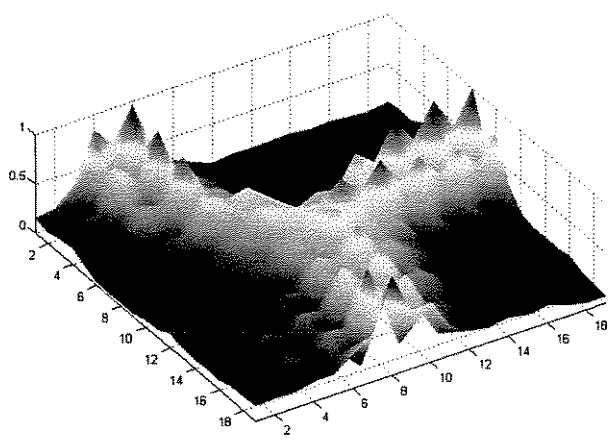


Figura 5.83: U-matrix (2D) para o SOM apresentado na figura 5.79.

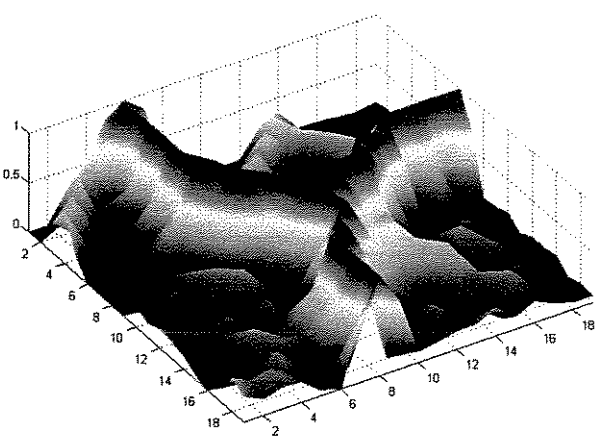


Figura 5.84: U-matrix (3D) para o SOM apresentado na figura 5.80.

Uma outra forma de usar o conhecimento de H no cálculo da *U-matrix* é fazer com que valores elevados em H, que correspondem a neurônios que estão agrupando bastante padrões, forcem a altura da *U-matrix* naquelas posições para baixo. Isto porque há maior probabilidade dos neurônios agruparem mais quando estão nos centros dos agrupamentos, que são os vales na superfície topográfica da *U-matrix*. Por outro lado, neurônios em posições com baixa densidade de agrupamentos geralmente estão em posições de

montanhas da superfície da *U-matrix*. Seja H' a matriz H escalonada no intervalo [0,1]. Uma saída para enfatizar tanto as bordas quanto os vales seria usar a informação de $(1\text{-}H'(i,j))^{\beta}$ no cálculo da *U-matrix*, por exemplo, multiplicando cada posição de neurônio da *U-matrix* por este valor, e fazendo a suavização de forma similar à feita na U-matrix, i.e., pela média das posições circunvizinhas. O expoente β serviria como parâmetro que controla a influência do fator $(1\text{-}H'(i,j))$ no cálculo da *U-matrix*.

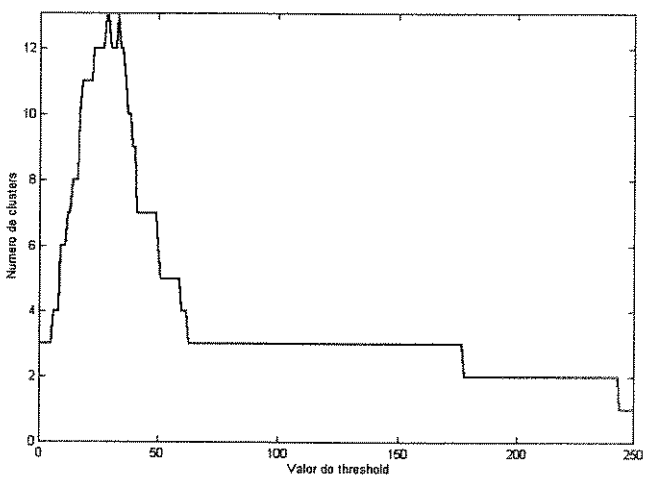


Figura 5.85: Gráfico do valor do limiar k versus N_{rc}^{k}. para a U-matrix apresentada na figura 5.82 e 5.84.

5.8 Sumário

Descrevemos a formulação matemática da *U-matrix* e propomos um método (o SL-SOM) de segmentação e rotulagem automática das regiões de neurônios, as quais formam modelos de representação distribuída dos protótipos como descrito no capítulo 2. Vários exemplos foram usados, mostrando a eficiência do SL-SOM detectando agrupamentos sem informações *a priori* da geometria das classes, ou do número destas. A estrutura se adapta ao problema, e não vice-versa, como ocorre nos algoritmos convencionais como o *k-means* e o método das misturas de Gaussianas. Além disto, uma breve interpretação do processo de agrupamento dos neurônios pelo SL-SOM foi abordada assim como maneiras de incorporar a informação de H (o histograma de vencimentos de padrões pelos neurônios) no cálculo da *U-matrix*. Nota-se que apenas agrupamentos conexos são obtidos pelo método SL-SOM, ao contrário do que ocorre no caso de misturas de densidades de probabilidades. Nesta última, regiões desconexas no espaço de atributos podem ocorrer devido ao uso da regra de Bayes e de estimativas das densidades de probabilidades, que utilizam a informação da matriz de covariância.

Capítulo 6

Hierarquias de Mapas Auto-organizáveis

Este capítulo estende o modelo SL-SOM com o objetivo de poder detectar, caso existam, subclasses de dados nos agrupamentos encontrados pelo SL-SOM. Uma estrutura de árvore dinâmica de redes SOM é proposta para tal objetivo, na qual regiões do mapa dão origem a outros mapas que são treinados apenas com subconjuntos de padrões que foram quantizadas pela região do mapa pai. O SL-SOM é aplicado a cada nível da árvore e métodos de verificação da importância de cada mapa em cada nível são propostos. Várias análises são feitas em conjuntos de dados diferentes e os resultados são mostrados assim como os comentários pertinentes.

6.1. Introdução

O capítulo anterior apresentou o SL-SOM, um método de segmentar e rotular automaticamente uma rede SOM treinada com um conjunto de padrões $\boldsymbol{X}$. Cada região de neurônios R_k do mapa possui um conjunto de neurônios associado ϑ_k, e podemos particionar $\boldsymbol{X}$ de acordo com a classe da região que possui o neurônio vencedor ou mais próximo a cada padrão $\boldsymbol{x} \in \boldsymbol{X}$. A operação pode ser resumida como,

$$\forall \boldsymbol{x}_i \in \boldsymbol{X}, \mathrm{C}(\boldsymbol{x}_i) = \Gamma(R_k) \quad | \quad \boldsymbol{m}_l \in R_k \quad \text{e} \quad \| \boldsymbol{m}_l - \boldsymbol{x}_i \| < \| \boldsymbol{m}_j - \boldsymbol{x}_i \|, \quad \forall j,\ j \neq l. \qquad (6.1)$$

onde $i = 1, 2, \ldots, n$, (n é o número de objetos no banco de dados $\boldsymbol{X}$), $\mathrm{C}(\boldsymbol{x}_i)$ simboliza a classe atribuída ao padrão $\boldsymbol{x}_i$, $\Gamma(R_k)$ é a classe atribuída à região R_k no momento da rotulagem pelo método RCC (ver seção 5.3), $k = 1, \ldots, K$, onde K é o número de regiões ou agrupamentos detectados pelo SL-SOM, e $\boldsymbol{m}_l$ é o neurônio vencedor para o padrão $\boldsymbol{x}_i$. Assim, o rótulo da região que detêm o neurônio vencedor é usada como classe para o padrão inserido. Fazendo a operação descrita pela equação (6.1) para cada padrão $\boldsymbol{x}_i \in \boldsymbol{X}$, obtemos K partições do conjunto de dados, $\boldsymbol{X}^K$. Cada subconjunto $\boldsymbol{X}^K$ é um agrupamento resultante do SL-SOM contendo n^K objetos.

Costa e Netto (1999c) apresentaram uma extensão ao modelo SL-SOM, na qual uma árvore dinâmica de redes neurais SOM é construída automaticamente a partir de um conjunto de dados $\boldsymbol{X}$. Denominando o mapa inicial como mapa raiz, cada região de neurônios R_k pode

dar origem a um novo mapa, denominado mapa filho da região k, M_k, que será treinado apenas com o subconjunto de dados X^K. Toda a operação efetuada no mapa raiz é feita igualmente no mapa filho, resultando em (sub)regiões que podem dar origem a outros mapas filhos. Não se especifica *a priori* o número de níveis da árvore nem o número de submapas para cada rede, e a dinâmica de todo o processo pode ser pensada como um particionamento recursivo do conjunto de dados. A estrutura final da árvore explica relacionamentos hierárquicos entre as várias classes de objetos detectados durante o processo.

O uso do SL-SOM habilita a detecção automática de regiões de cada mapa mantendo o conceito de representação distribuída de protótipos, advogada nesta tese. Outras redes competitivas hierárquicas foram propostas, como descrito no capítulo 4, porém, quando derivadas do SOM possuem estrutura fixa *a priori* (por exemplo a HSOM), necessitando de rotulação manual em cada estágio, ou cada nó da árvore é um neurônio (por exemplo a TS-SOM), que para efeitos de agrupamentos e classificação automática de padrões tem o mesmo problema dos métodos que usam a abordagem *centróide mais próximo*.

Em seu mais recente livro, Kohonen (1997a, página 159) brevemente toca no assunto de redes hierárquicas, o qual replicamos um trecho logo a seguir.

> *"One objective in SOM research has been to construct structured maps using elementary SOMs as modules. Such constructs are still at an elementary stage of development".*

Demonstramos, neste capítulo, que o uso recursivo do SL-SOM numa estrutura hierárquica irá produzir resultados bastante satisfatórios.

6.2. Um pouco sobre árvores de decisão e estruturas de árvores

Árvores de decisão são comuns em áreas como reconhecimento de padrões e aprendizado por máquina, porém em geral aplica-se apenas em problemas de classificação supervisionada. As estruturas mais simples são as árvores binárias que efetuam decisão Booleana ao longo de eixos (ou variáveis), i.e., cortes ortogonais nas coordenadas do espaço de atributos. Uma comparação entre redes neurais do tipo *multilayer perceptron* e árvores de decisão lineares foi efetuada por Park (1994). O autor descreve que os resultados em conjuntos de dados sintéticos e reais foram similares, em ambas as técnicas, e defende o

método de árvores pelo fato de sua simplicidade de interpretação, de como foi efetuada a decisão, além de requerer menor tempo para treinamento e execução.

Brodley & Utgoff (1995) descrevem árvores de decisão multivariadas nas quais cada nó efetua decisão baseando-se em mais de uma variável. Os autores concluíram que a delimitação das regiões das classes foi melhor representada, i.e., de forma mais sucinta, e consegui-se obter árvores menores, o que também implica em menor tempo de processamento. Uma boa introdução a técnicas de construção de árvores de decisão e classificadores estruturados como árvores pode ser vista em Ripley (1996, capítulo 7).

Basicamente, árvores são estruturas gráficas, i.e., um tipo especial de grafo, onde há um nó ou elemento denominado raiz, que geralmente está no topo da árvore. Nós filhos são conectados por arcos, e nós que não possuem filhos são denominados nós folhas. Por definição, há um caminho único entre dois nós quaisquer de uma árvore. O nível de um nó em uma árvore pode ser definido atribuindo-se nível zero ao nó raiz e qualquer outro nó na árvore é um nível a mais que o nível do nó de seu pai. Profundidade pode ser definida como o nível máximo de qualquer folha na árvore. O tipo de árvore mais simples é a árvore binária onde apenas dois filhos são permitidos para cada nó. Algoritmos para geração e controle de estruturas de árvores são bastante conhecidos. Uma boa referência é Tenenbaum et al. (1995).

No nosso caso, cada nó da árvore será um mapa ou rede de Kohonen, com parâmetros definidos a partir dos parâmetros do mapa pai e também dos dados que serão utilizados para treinamento. Cada mapa terá um endereço na árvore, e descreveremos neste capítulo esquemas de geração automática da árvore, que é dinâmica, i.e., em princípio não sabemos qual será a profundidade da árvore. Também não haverá restrições em relação ao número de filhos de um nó da árvore, e a estrutura gerada pode ser desbalanceada.

6.3. Hierarquias de mapas SOM

Cada região rotulada em um mapa no nível η da árvore[1], R_k^{η}, pode dar origem a um novo mapa em um nível $(\eta+1)$, denominado mapa filho da região k, M_k^{η}, que será treinado apenas com o subconjunto de dados X_k^{η}. A figura 6.1 ilustra uma região de neurônios (parte superior em destaque no centro do mapa) e seu mapa filho correspondente (parte inferior). Como veremos adiante, o mapa filho pode herdar características da região que lhe deu origem.

[1] O mapa raiz está no nível zero da árvore dinâmica.

A extensão do SL-SOM tem por objetivo descobrir e representar subclasses de forma hierárquica. A figura 6.2 ilustra o exemplo onde o mapa raiz teve três regiões de neurônios. Dedicando atenção apenas a um dos mapas filho, o mapa 2 do nível 1 (que está no centro), vemos que este detectou quatro regiões, dando origem a mais quatro subredes. A subrede 2 do nível 2 detectou duas regiões, originando dois mapas filhos no nível 3.

Dois fatores básicos controlam a dinâmica de crescimento da árvore: (1) o número de sub-mapas para um dado mapa em um dado nível; e (2) o processo de parada de crescimento.

O algoritmo TS-SL-SOM (*Tree-Structured Self-Labeled SOM*) é apresentado a seguir.

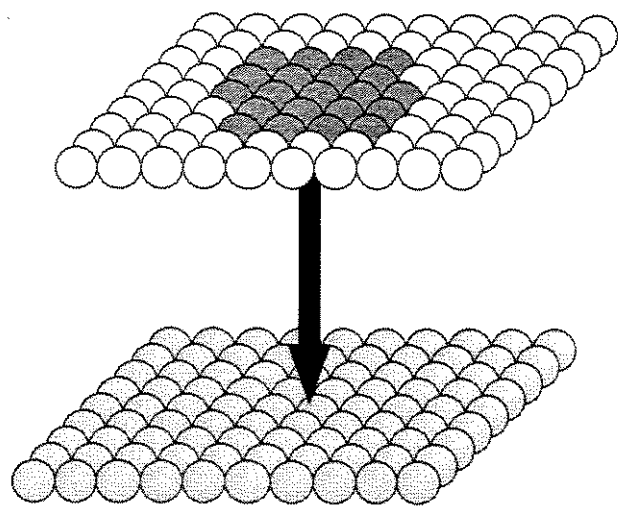


Figura 6.1: Uma região de neurônios e seu mapa filho.

6.3.1 O algoritmo TS-SL-SOM

Antes de iniciar o processo deve-se escolher, como no caso do SOM e do SL-SOM, parâmetros como o tamanho e dimensão do mapa, tipo de vizinhança, funções $\alpha(t)$ e $h(d, t)$, número de iterações e forma de inicialização dos pesos.

1. Seja o nível da árvore atual $\eta = 0$. O conjunto de dados X^{η} *para* $\eta = 0$ é o próprio conjunto de dados X.

2. Efetue o treinamento do mapa M^{η} com o conjunto de dados X^{η}.

3. Seja K^{η} o número de regiões ou agrupamentos detectados no mapa M^{η}. Rotule os neurônios das regiões de acordo com o SL-SOM e classifique os padrões do conjunto de dados X^{η}, o que resulta em uma partição de X^{η} em K^{η} subconjuntos $X(K^{\eta}; M^{\eta})$.

4. Cada região K^{η} do mapa M^{η} dá origem a um submapa no nível $(\eta + 1)$, subordinado ao mapa pai, o qual denominaremos $M^{\eta+1}(K^{\eta})$. O tamanho deste mapa é feito proporcional ao tamanho do mapa pai e ao tamanho do subconjunto de dados $X(K^{\eta}; M^{\eta})$ que será usado para treiná-lo. Seja ζ o tamanho do mapa pai (em número de neurônios) e q_k a fração do número de padrões do subconjunto de dados $X(K^{\eta}; M^{\eta})$ pelo número de padrões do conjunto de dados usado no mapa pai, X^{η}. Assim,

$$q_{K^{\eta}} = \frac{\left|X(K^{\eta}; M^{\eta})\right|}{\left|X^{\eta}\right|}$$

onde $|\cdot|$ representa a cardinalidade do conjunto de dados. O tamanho do mapa filho $M^{\eta+1}(K^{\eta})$, será

$$\zeta_{K^{\eta}}^{\eta+1} = (q_{K^{\eta}})^{\beta} \cdot \zeta^{n}$$

onde o valor de β usado nesta tese foi 0.3.

5. Treine o mapa $M^{\eta+1}(K^{\eta})$ com o conjunto de dados $X(K^{\eta}; M^{\eta})$, este último pode ser re-normalizado, de forma que possamos focalizar atenção no subconjunto de dados. Rotula-se os neurônios do mapa $M^{\eta+1}(K^{\eta})$ da mesma forma como no mapa raiz, igualmente gerando-se subconjuntos de dados.

6. Caso um submapa não possua partições ele é eliminado, e o subconjunto de dados fica representado pelo região do mapa pai correspondente. Isto ocorre quando o sistema não consegue detectar subgrupos nos dados, por exemplo, quando os objetos são provenientes de uma única população, como uma Gaussiana ou uma distribuição uniforme.

7. Repita os passos 4-6 até que a estrutura em árvore estabilize, i.e., não consiga adicionar nem eliminar submapas.

Note que o treinamento é feito localmente em cada submapa com uma fração do conjunto de dados usado no mapa pai. Os tipos de normalização de dados permitidos são diversos, porém, os mais comuns são normalização para média zero e variância 1, ou escalonamento linear, que transforma a faixa de valores [mínimo, máximo], de cada atributo, em [0, 1], por exemplo.

No passo 4, a escolha da função para o tamanho dos mapas filhos teve objetivo de permitir que o tamanho dos submapas diminuísse à medida que avança-se no nível da árvore, η, levando em consideração a proporção do número de padrões do mapa pai que cada região ficou responsável. O valor $\beta = 0.3$ foi motivado pela razão de que é importante decrescer o tamanho dos mapas filhos em relação ao mapa pai, porém isto deve ser feito de forma suave, até porque não é interessante ter um mapa muito pequeno, pois irá dificultar o processamento das imagens geradas (*U-matrix*) no SL-SOM. Este valor foi empiricamente escolhido após várias simulações em conjuntos de dados artificiais, porém, não é crítico, inclusive o tamanho dos mapas filhos pode ser escolhido de formas diferentes, por exemplo, no mesmo tamanho do mapa pai. Uma das motivações do decrescimento do tamanho dos submapas inclui economia de memória, o que no futuro espera-se não constituir mais um problema importante.

Note que cada mapa é independente em relação aos mapas do mesmo nível da árvore. Nesta tese, o nível da árvore η é incrementado apenas quando todos os mapas em um determinado nível foram segmentados e rotulados, porém não há nenhum problema em focalizar atenção em um ramo da árvore até que não possa mais gerar subdivisões para então iniciar o processo nos submapas de mesmo nível. Em relação ao passo 6, vemos que um determinado mapa só será mantido na árvore se possuir no mínimo duas regiões. Caso contrário todo o subconjunto de dados utilizado para treinamento deste mapa fica representado pela região do mapa pai responsável pela sua geração.

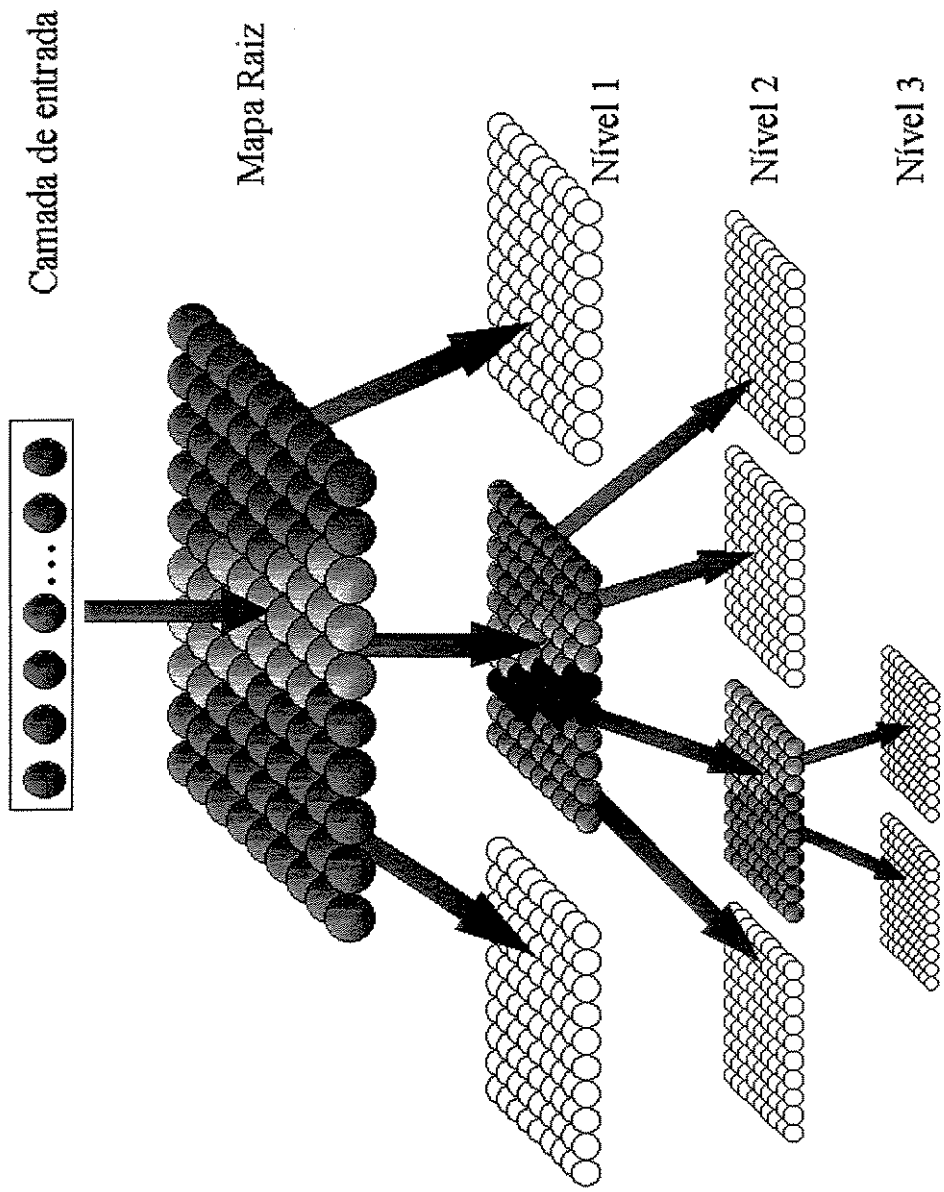


Figura 6.2: Ilustração do processo de geração da árvore dinâmica pelo TS-SL-SOM.

6.3.2 Critérios de parada nos ramos da árvore

Um dos principais problemas no TS-SL-SOM é a determinação da informação de que um dado mapa não possui sub-mapas (passo 6), e desta forma a tentativa de geração de mapas filhos deve ser anulada.

Para sermos capazes de criar uma estratégia de parada do TS-SL-SOM, devemos inicialmente definir o tipo de estruturas que estamos buscando. Um agrupamento deveria possuir duas características básicas: isolação externa e coesão interna (Gordon, 1981). Estas duas características estão relacionadas com a configuração dos objetos dos agrupamentos: nenhum objeto deveria ser mais próximo de outro objeto de um outro agrupamento do que a um objeto do mesmo agrupamento. Isto nem sempre é verdadeiro, por exemplo, no caso de dois agrupamentos alongados em paralelo.

Em geral, assume-se que dados provenientes de (apenas) uma distribuição uniforme não formam agrupamentos. Na realidade, ou consideramos que os objetos pertencem a um grande agrupamento, ou cada objeto por si só é um agrupamento diferente. Como alguns dos métodos estatísticos consideram a forma Gaussiana para modelo dos agrupamentos encontrados, deveríamos também ser capazes de detectar quando há apenas uma classe proveniente de uma normal p-variada.

A seguir, mostraremos exemplos de simulações com o SL-SOM para densidades Gaussianas e uniformes, com o objetivo de ilustrar os problemas da determinação de um critério de parada para o TS-SL-SOM.

6.3.2.1. Distribuição uniforme

A figura 6.3-a ilustra a configuração de neurônios de um SOM de tamanho 10×10 após inicialização linear, enquanto que a figura 6.3-b mostra o resultado do treinamento do mesmo mapa após 1000 iterações do algoritmo em lote, a partir de um conjunto de dados gerado artificialmente. O conjunto contém 5000 pontos de uma densidade uniforme bidimensional. Note que há uma contração nas bordas do SOM, e as distâncias dos neurônios situados nas bordas aos vizinhos não situados em bordas é menor do que as distâncias entre os neurônios no centro do *grid*. Note que há homogeneidade nas distâncias entre todos os neurônios não situados em bordas, o que reflete a capacidade do SOM em estimar a distribuição uniforme dos dados.

O histograma de vencedores é apresentado na figura 6.4. Note, inclusive pela figura 6.3-b, que os neurônios da periferia do *grid* possuem maior número de padrões associados, também por problemas de distorção causada pelo tamanho finito da rede, i.e., bordas da rede. Quanto maior o tamanho do mapa, menor o efeito dos contornos (Kohonen, 1997a).

A *U-matrix* é apresentada na figura 6.5. Note o platô de distâncias praticamente iguais no centro do mapa, enquanto em toda periferia há buracos derivados do efeito dos contornos

do mapa (ver figura 6.3-b). A figura 6.6 mostra a *U-matrix* da configuração de neurônios apresentada na figura 6.3-b considerando apenas os neurônios que não fazem parte da periferia do mapa, i.e., não estão em posições de borda. Note que as distâncias são bem próximas entre si, no caso, a máxima distância foi 0.1036 enquanto que a mínima foi 0.0859, com média de 0.0939 e desvio padrão de 0.0027. Note que o coeficiente de variação, dado pelo quociente do desvio padrão pela média, foi bastante baixo, 0.0288.

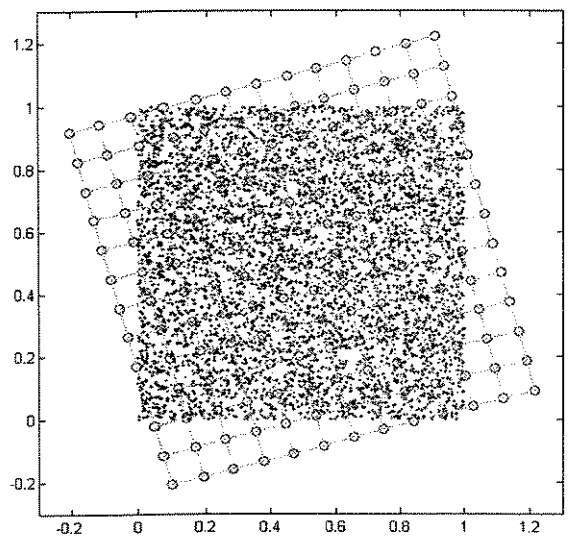


Figura 6.3-a: Grid de neurônios de um mapa com tamanho 10 × 10, e os dados, após inicialização linear.

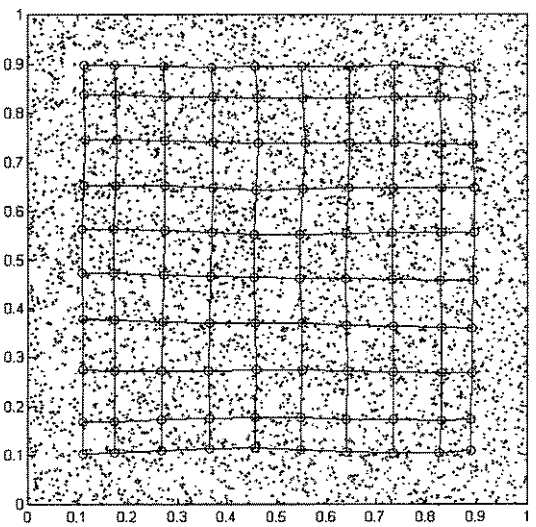


Figura 6.3-b - Grid de neurônios, e dados, do mapa de tamanho 10 × 10 após 1000 iterações do algoritmo em lote, treinado a partir de um conjunto de dados baseado em uma distribuição uniforme bidimensional.

	1	2	3	4	5	6	7	8	9	10
1	99	57	62	60	66	55	67	74	67	89
2	60	36	37	43	41	33	32	37	21	60
3	61	35	49	41	42	47	52	52	41	63
4	65	34	41	42	41	36	44	45	34	78
5	69	37	42	49	52	38	47	50	33	61
6	66	37	39	46	44	48	45	44	31	66
7	66	32	50	36	39	40	38	41	39	66
8	56	36	41	42	41	47	53	55	36	75
9	54	30	42	32	37	32	23	35	31	53
10	99	68	63	71	69	73	65	76	54	101

Figura 6.4 - Histograma de vencedores para o SOM apresentado na figura 6.3-b

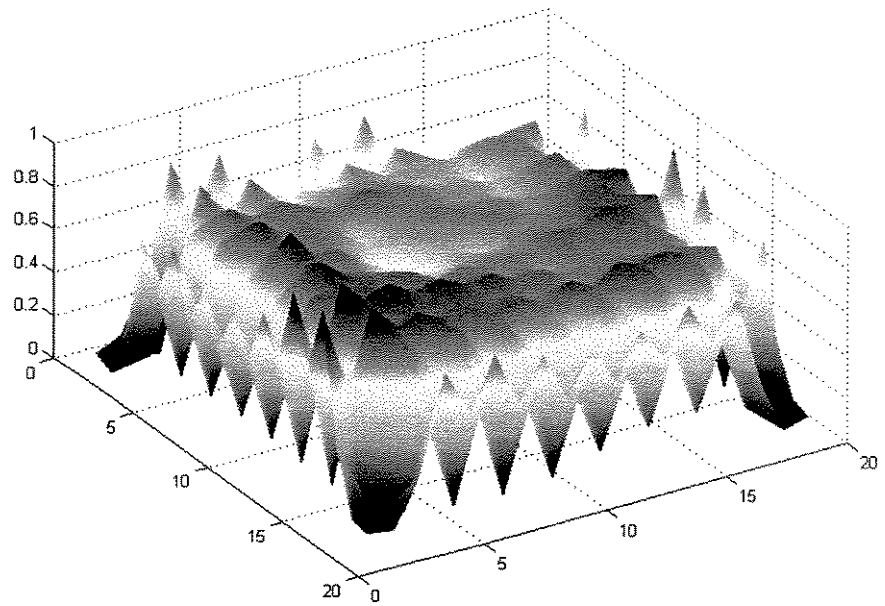


Figura 6.5 - U-matrix para o SOM apresentado na figura 6.3-b

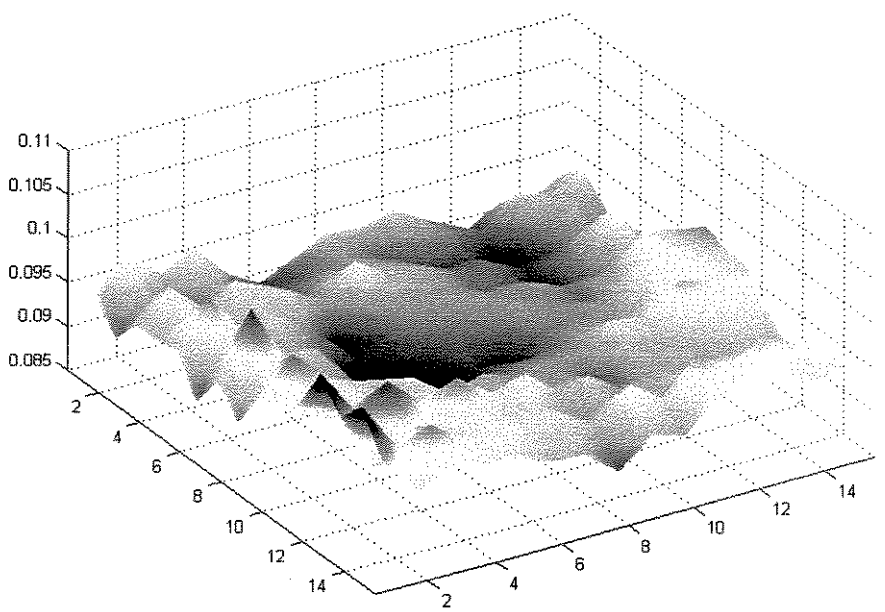


Figura 6.6 - U-matrix para o SOM apresentado na figura 6.3, desconsiderando os neurônios da periferia do grid.

A figura 6.7 ilustra o número de regiões conectadas *versus* o valor de limar para a *U-matrix* apresentada na figura 6.6. Note que há um deslocamento para a direita do gráfico, em relação a um gráfico de um problema em que há realmente agrupamentos. Para os valores de limiares de 0 a 68, o número de regiões estáveis é 1, indo para 2 no intervalo 69 a 84, e depois 3 no intervalo 85 a 106. Este valor cai para 2 no intervalo 107 a 111, e depois para 1, entre 112 a 118. Note uma oscilação, no valor de limiar 119 o número de regiões volta a 2 logo caindo para 1 novamente, e voltando a 2 no intervalo 124 a 140, depois indo de 141 até 255 no valor 1.

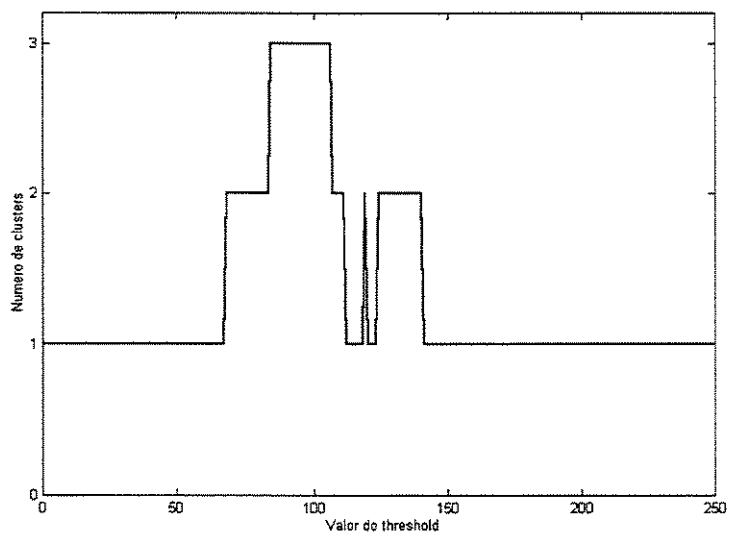


Figura 6.7 - Gráfico do número de regiões conectadas versus o valor de limar para a U-matrix apresentada na figura 6.6.

Se considerarmos os neurônios da borda da rede teremos um gráfico bastante diferente do apresentado na figura 6.7 (ver figura 6.8). Note que poderíamos considerar a existência de quatro agrupamentos, como sugere a figura 6.8, porém, estes agrupamentos são devidos a problemas com distorções de bordas do SOM. Estes quatro agrupamentos são derivados dos quatro vértices do mapa, como podemos ver na figura 6.9, que é a imagem bidimensional da *U-matrix* apresentada na figura 6.5.

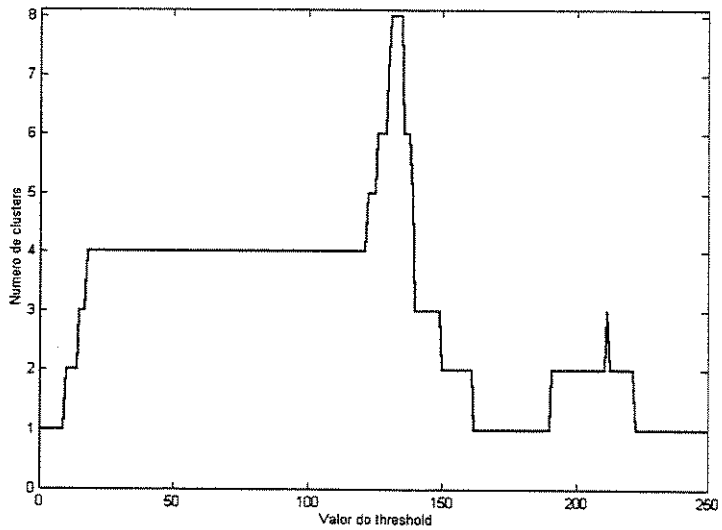


Figura 6.8 - Gráfico do número de regiões conectadas versus o valor de limiar para a U-matrix apresentada na figura 6.5.

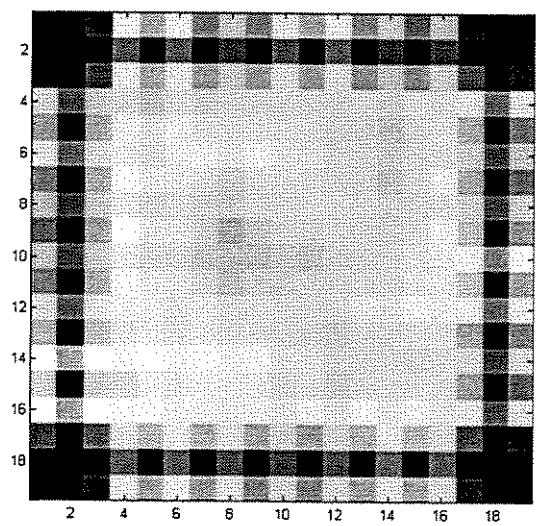


Figura 6.9 - Imagem da U-matrix apresentada na figura 6.5.

6.3.2.2. Distribuição Gaussiana

Testes foram efetuados com distribuições Gaussianas de forma a tentar entender o resultado em um mapa da projeção de um ou mais agrupamentos oriundos de dados gerados por densidades de probabilidades Gaussianas.

A figura 6.10-a ilustra a configuração de neurônios de um SOM com topologia bidimensional e tamanho 12×12 após inicialização linear, enquanto que a figura 6.10-b ilustra a configuração resultante após 1000 iterações usando o algoritmo em lote. O conjunto de dados usado foi uma população de uma normal com média zero e desvio padrão 1. Note que neste caso usamos um escalonamento linear, de forma que os valores estão contidos no intervalo [0, 1]. Note, como era esperado, que há maior concentração de neurônios no centro, onde a densidade da distribuição é mais intensa.

Diferentemente do caso da distribuição uniforme, neste caso, alguns neurônios da periferia do mapa possuem distâncias aos neurônios adjacentes maiores que as distâncias entre os neurônios do centro do mapa. Entretanto, nota-se a acomodação da grade elástica do SOM aos dados, havendo uma certa distorção da grade, que se deve à natureza discreta do problema. Isto pode ser diminuído com o aumento do número de padrões do conjunto de dados.

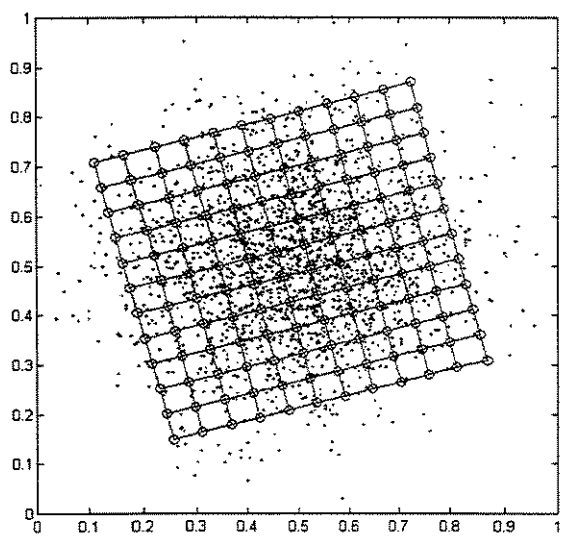


Figura 6.10-a: Grid de neurônios de um mapa com tamanho 12 × 12, e os dados, após inicialização linear.

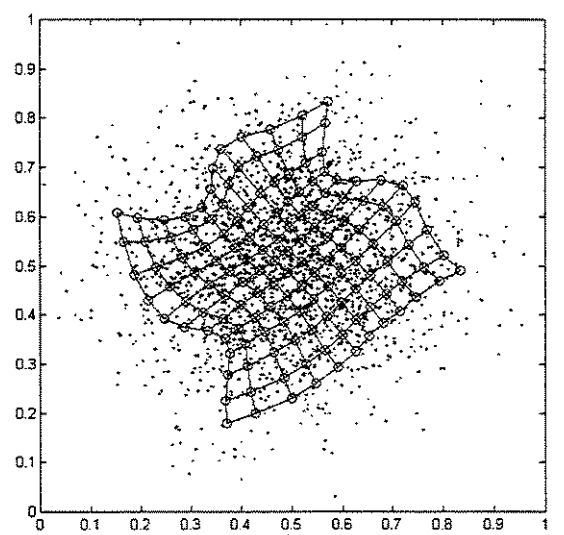


Figura 6.10-b - Grid de neurônios, e dados, do mapa de tamanho 12 × 12 após 1000 iterações do algoritmo em lote, treinado a partir de um conjunto de dados baseado em uma distribuição Gaussiana bidimensional.

Note no histograma de vencedores na figura 6.11 que, igualmente ao caso da distribuição uniforme, temos uma certa homogeneidade de valores, não apresentando nenhuma região de baixa densidade de padrões (sendo mapeados nos neurônios) separando duas ou mais regiões de elevada densidade de padrões. No centro do mapa há maior número de padrões

sendo mapeados, como esperávamos, devido à propriedade do SOM em aproximar a densidade do espaço dos dados.

Pelo fato das distâncias entre os neurônios no centro serem menores, espera-se uma *U-matrix* cuja superfície topográfica apresente um grande *vale* no centro rodeado por elevações. A *U-matrix* pode ser vista na figura 6.12.

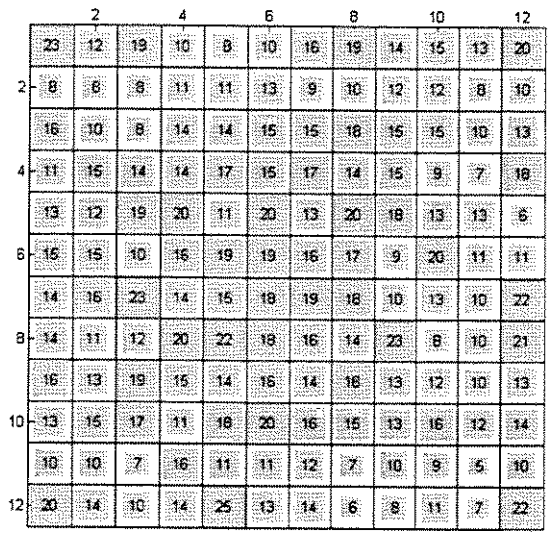


Figura 6.11 - Histograma de vencedores para o SOM apresentado na figura 6.10-b

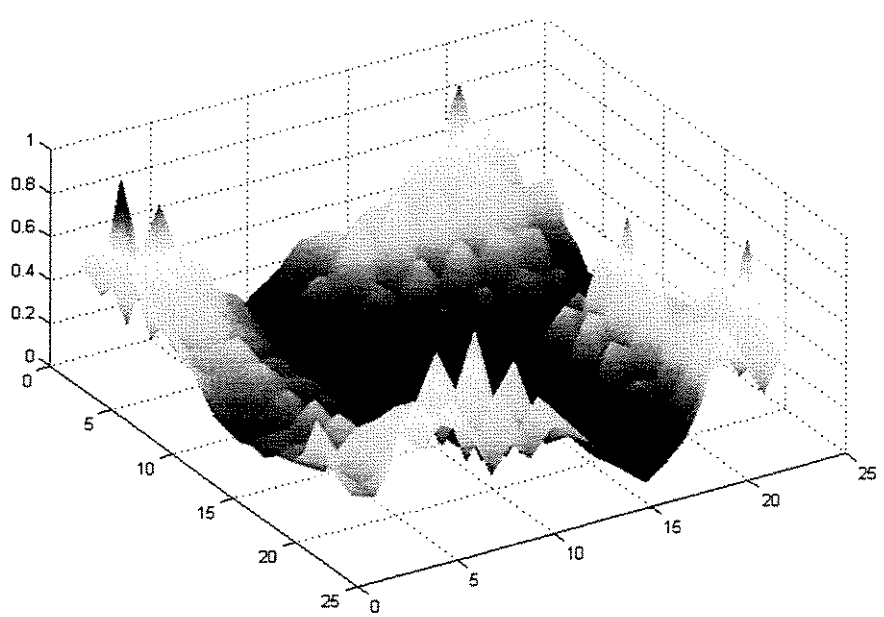


Figura 6.12 - U-matrix para o SOM apresentado na figura 6.10-b.

Note o comportamento do número de regiões conectadas (figura 6.13) em relação ao valor do limiar da *U-matrix*. Note que, já para o nível de cinza 50, o número de agrupamentos cai para 1, e depois volta para 2, retornando para 1, oscilando, devido a buracos na *U-matrix*.

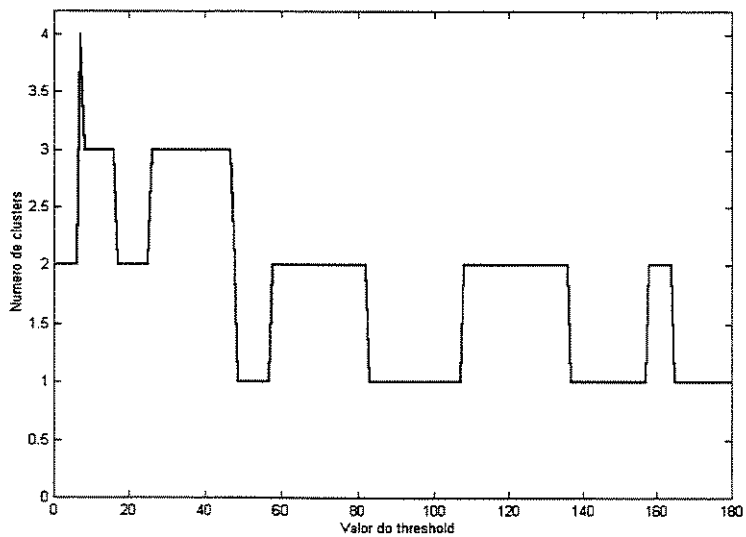


Figura 6.13 - Gráfico do número de regiões conectadas versus o valor de limiar para a U-matrix apresentada na figura 6.12.

Desconsiderando os neurônios das bordas, obtemos a *U-matrix* apresentada na figura 6.14. Note que efetivamente trabalhamos com uma versão suavizada da *U-matrix*, como foi descrito no capítulo 5. O resultado da suavização é apresentado na figura 6.15.

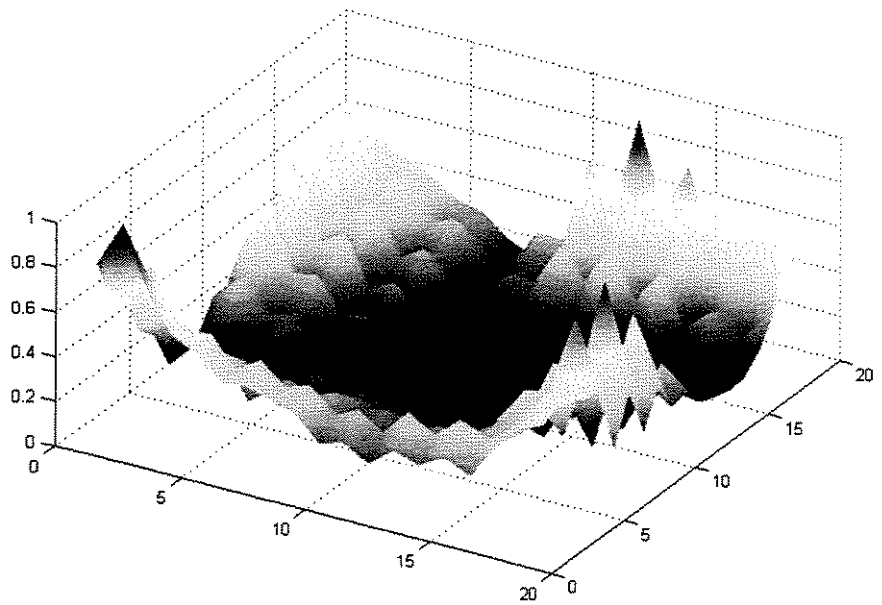


Figura 6.14 - U-matrix para o SOM apresentado na figura 6.10-b, desconsiderando os neurônios da periferia do grid.

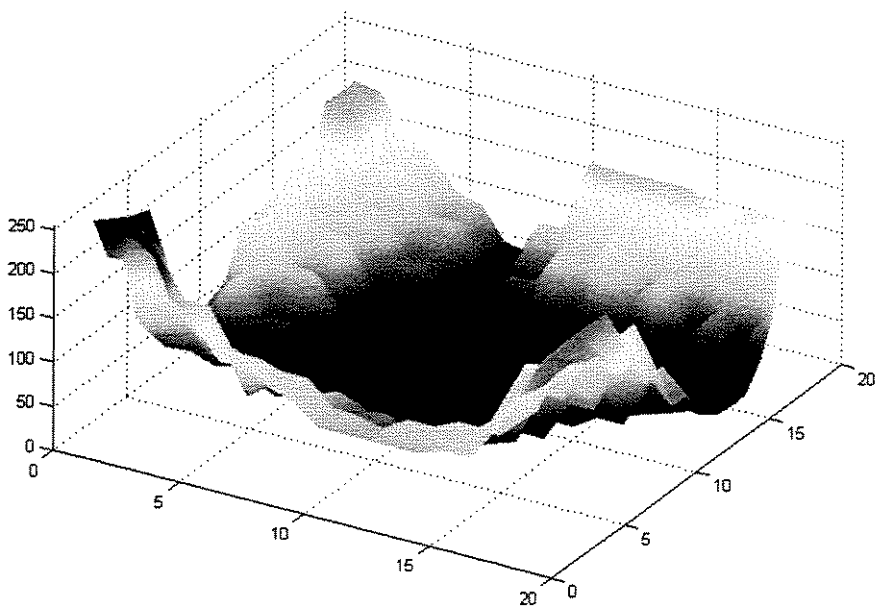


Figura 6.15 - U-matrix apresentada na figura 6.13 após suavização.

O gráfico do número de regiões conectadas *versus* limiar da *U-matrix* para o caso em que não consideramos os neurônios na periferia do mapa é apresentado na figura 6.16. Note, em relação à figura 6.13, que praticamente apenas um agrupamento foi detectado, havendo apenas uma rápida transição de 1 para 2 no valor de limiar 86. Em todos os outros níveis de cinza foi obtida apenas uma região conectada, o que implica em apenas um agrupamento.

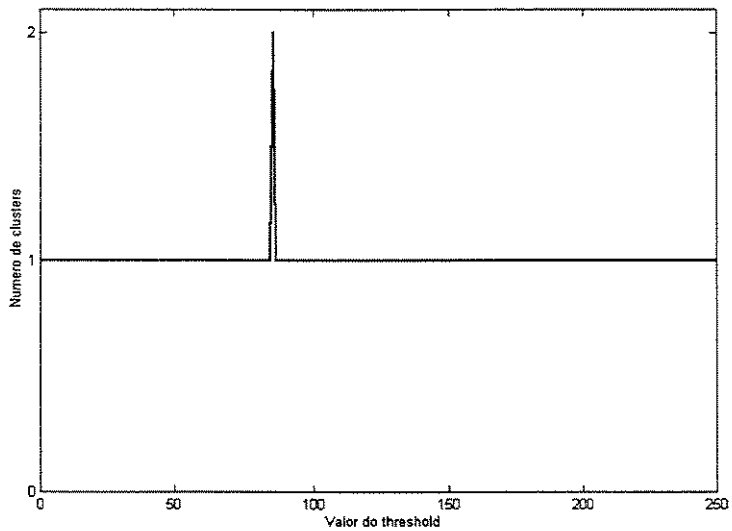


Figura 6.16 - Gráfico do número de regiões conectadas versus o valor de limiar para a U-matrix apresentada na figura 6.15.

6.3.3 Detecção de centros de ativação no mapa

Um outro fator para determinação da existência de agrupamentos de neurônios no mapa é considerar os centros de ativação do mapa.

Considere a apresentação de um padrão $\boldsymbol{x}$ à rede treinada, por exemplo, ao SOM apresentado na figura 6.3. Calculando a ativação de todos os neurônios em relação ao padrão $\boldsymbol{x}$ teremos um vencedor, um segundo vencedor, e assim por diante. Como foi descrito no capítulo 3, podemos considerar a ativação do mapa para um padrão como sendo a distância Euclidiana ou o produto interno[2] entre o padrão $\boldsymbol{x}$ e cada neurônio $\boldsymbol{m}_i$. No primeiro caso temos um valor que representa uma dissimilaridades enquanto que no segundo uma similaridade. O neurônio vencedor c (ver capítulo 3) é o que apresenta a menor (maior) dissimilaridade (similaridade) com o padrão $\boldsymbol{x}$. A figura 6.17 ilustra as ativações de todos os neurônios, para um dado padrão $\boldsymbol{x}$ = { 0.4546, 0.5522}, para o mapa apresentado na figura 6.3.

[2] Se os pesos estão com norma 1.

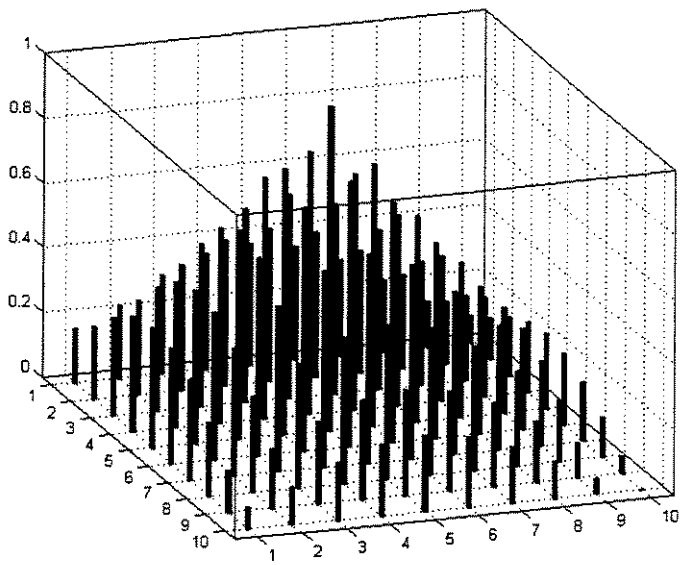


Figura 6.17 - Ativações para todos os neurônios do mapa apresentado na figura 6.3, para um dado padrão $\boldsymbol{x}$ = { 0.4546, 0.5522}.

Note que como usamos a distância Euclidiana como função de ativação, o neurônio vencedor teria ativação mínima enquanto que o neurônio mais longe do padrão teria ativação máxima. O gráfico apresentado na figura 6.17 apresenta uma informação de similaridade do padrão com cada neurônio $\boldsymbol{m}_i$ do mapa, dado pela relação

$$Ativ(\boldsymbol{m}_i) = \exp\left(-\left\|\boldsymbol{x} - \boldsymbol{m}_i\right\|\right)$$

onde $\|\cdot\|$ representa a distância Euclidiana.

Desta forma, $Ativ(\boldsymbol{m}_i)$ será 1 quando houver coincidência entre o padrão $\boldsymbol{x}$ e o neurônio $\boldsymbol{m}_i$. Quanto mais distante $\boldsymbol{m}_i$ estiver do padrão, menor será sua ativação. Note nas figuras 6.18 e 6.19 a bolha de ativação ao redor do neurônio vencedor c cujas coordenadas são (5, 5). A figura 6.18 ilustra a mesma informação da figura 6.17 de forma planar, enquanto que a figura 6.19 ilustra o percentual de ativação dos neurônios do mapa para o padrão $\boldsymbol{x}$ em relação à ativação do neurônio vencedor c. Note que houve um escalonamento linear nas ativações, i.e., a máxima ativação foi considerada 1 (100%) enquanto a mínima foi para zero. Todos os outros valores foram interpolados linearmente entre estes extremos.

Esta informação pode ser vista igualmente na forma de uma superfície 3D ou uma imagem. As figuras 6.20 e 6.21 ilustram a ativação em relação ao padrão $\boldsymbol{x}$ inserido no mapa. A informação e análise é equivalente às das figuras 6.18 e 6.19.

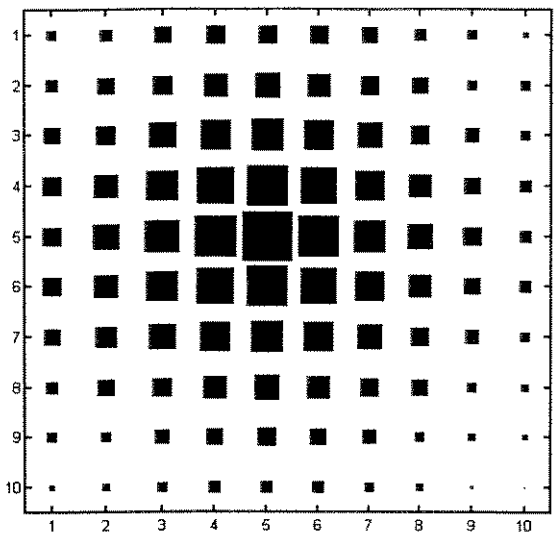


Figura 6.18: Ilustração (planar) das ativações para todos os neurônios do mapa apresentado na figura 6.3, para um dado padrão x = { 0.4546, 0.5522}.

	1	2	3	4	5	6	7	8	9	10
1	17	22	31	36	37	35	30	22	14	9
2	22	29	38	45	47	45	38	29	19	13
3	30	38	50	59	63	59	49	38	26	18
4	35	45	68	74	81	73	59	45	30	21
5	37	47	64	82	100	81	63	48	32	23
6	35	45	61	75	82	74	60	45	31	22
7	31	39	51	60	64	60	50	38	26	18
8	23	29	38	45	47	45	38	29	19	12
9	13	18	25	30	32	31	26	19	11	5
10	7	11	18	22	23	22	17	11	5	0

Figura 6.19: Percentual das ativações dos neurônios em relação ao padrão x = { 0.4546, 0.5522}, considerando 100 a ativação do neurônio vencedor e 0 o neurônio mais distante.

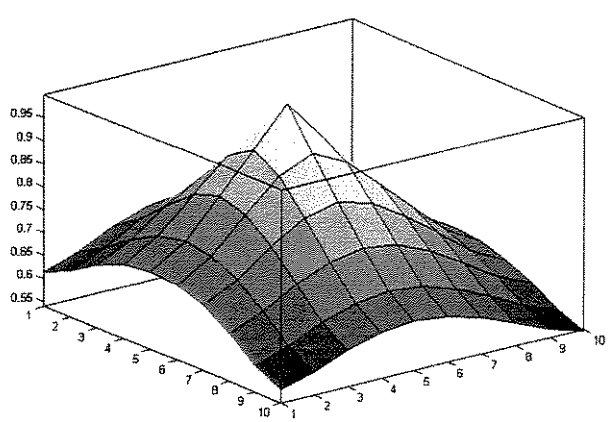


*Figura 6.20 - Ativação do mapa para o padrão **x** inserido. Visualização como superfície*

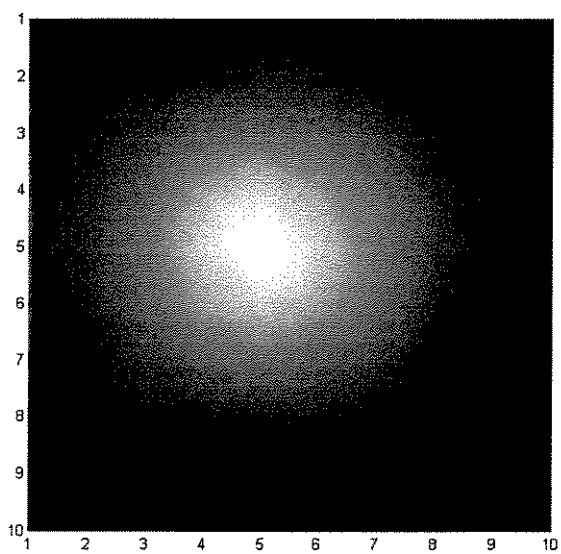


*Figura 6.21 - Ativação do mapa para o padrão **x** inserido. Visualização como imagem*

A ativação média em cada neurônio *i* do mapa (Λ_i) pode ser definida como a média de todas as ativações considerando o conjunto de dados *X*.

$$\Lambda_i = \frac{1}{n}\sum_{l=1}^{n}\exp\left(-\left\|\boldsymbol{x}_l - \boldsymbol{m}_i\right\|\right)$$

onde cada i denota uma posição no mapa. O valor Λ_i é, portanto, uma média da similaridade de todos os padrões $\boldsymbol{x}_l \in X$ a cada neurônio $\boldsymbol{m}_i$ do mapa.

A ativação média para um mapa, Λ_{mapa}, é a imagem das ativações médias de todos os neurônios i que fazem parte do mapa. Para o SOM apresentado na figura 6.3, Λ_{mapa} é mostrada na figura 6.22. A figura 6.23 ilustra a mesma informação da figura 6.22 na forma de imagem em níveis de cinza. Nota-se a existência de apenas um centro de ativação, que é definido como um máximo regional, i.e., o neurônio i será um centro de ativação caso $\Lambda_i > \Lambda_j$, $j = 1, 2, \ldots, 8$, $j \in N(i)$, onde $N(i)$ denota o conjunto de neurônios vizinhos ou adjacentes ao neurônio i. Ou seja, considerando a imagem da ativação média para cada neurônio do mapa, estamos buscando máximos regionais. Isto pode ser feito de forma simples deslocando uma máscara de tamanho 3×3 na imagem, do canto superior esquerdo ao canto inferior direito da imagem, e comparando a ativação média do neurônio central da máscara com os neurônios vizinhos. Esta busca pelos picos de ativação foi sugerida por Su et al. (1997). A figura 6.24 ilustra o resultado após detecção do único pico ou centro de ativação encontrado neste problema (SOM apresentado na figura 6.3), que foi no neurônio de coordenadas (5, 6) do mapa.

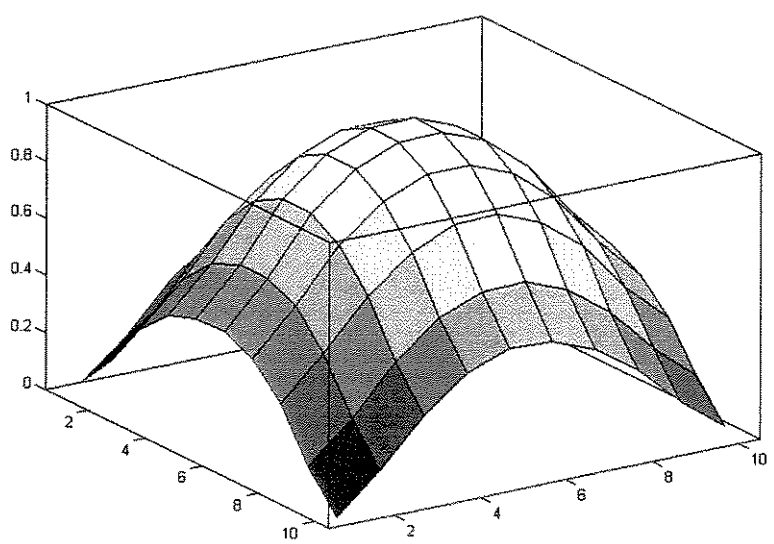


Figura 6.22 - Ativação média para o SOM apresentado na figura 6.3.

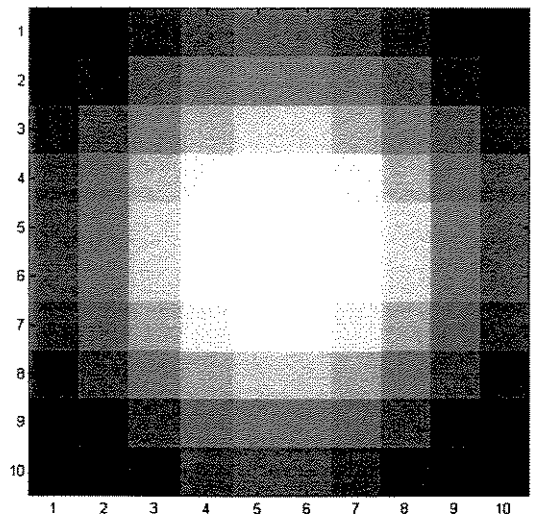


Figura 6.23 - Imagem da ativação média para o SOM apresentado na figura 6.3.

O centro de ativação encontrado está na posição (5, 6) do mapa, onde a primeira coordenada explicita a linha e a segunda a coluna.

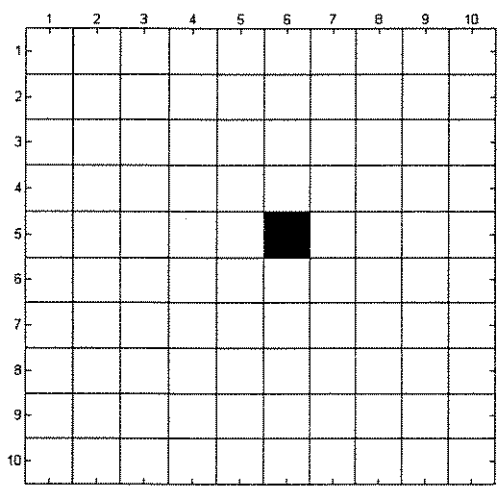


Figura 6.24 - Único pico ou centro de ativação encontrado para o SOM apresentado na figura 6.3.

Para o SOM treinado com dados provenientes de uma distribuição Gaussiana, ver figura 6.10, a ativação média é mostrada na figura 6.25. Considerando a ativação média do mapa, como descrito anteriormente, obtemos apenas um centro de ativação, localizado no neurônio com posição (7, 7). As figuras 6.26(a) e 6.26(b) ilustram outras visualizações da ativação do mapa para este problema.

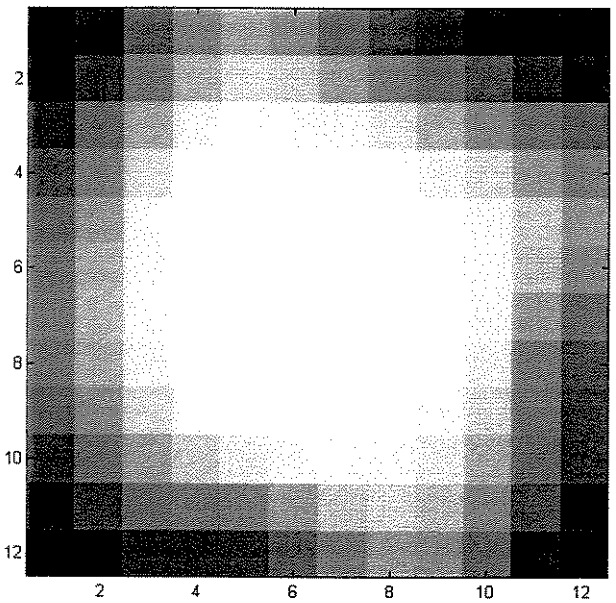


Figura 6.25 - Imagem da ativação média para o SOM apresentado na figura 6.10.

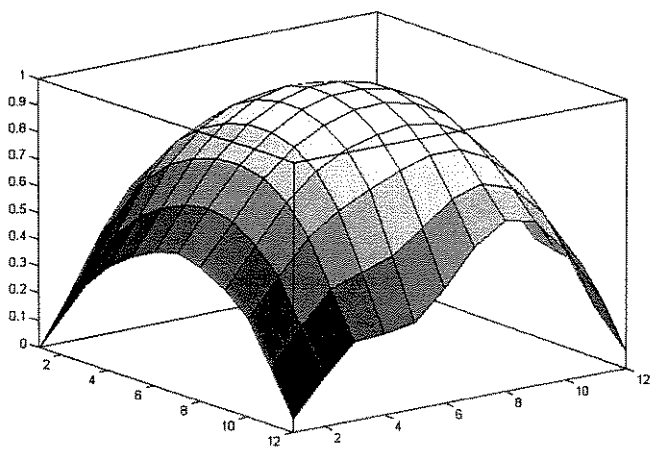


Figura 6.26 (a) - Ativação média acumulada para o SOM apresentado na figura 6.10 - superfície.

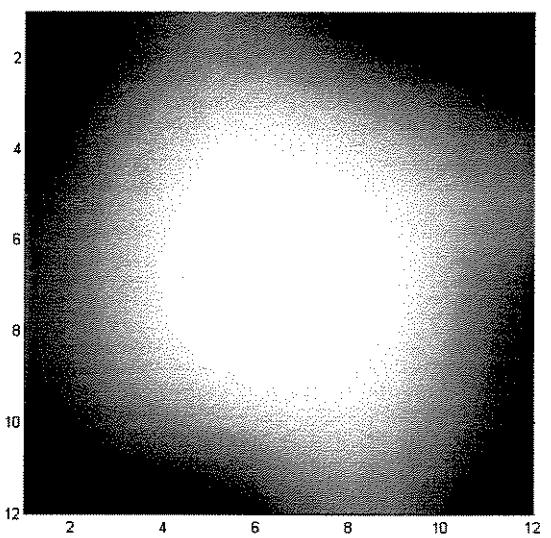


Figura 6.26 (b) - Ativação média acumulada para o SOM apresentado na figura 6.10 - imagem.

Para o SOM apresentado na figura 3.20, a ativação do mapa é apresentada nas figuras 6.27 e 6.28. Nota-se a existência de três centros de ativação, que foram detectados pela máscara 3×3 na imagem de ativações médias do mapa. Os centros de ativação são apresentados na figura 6.29.

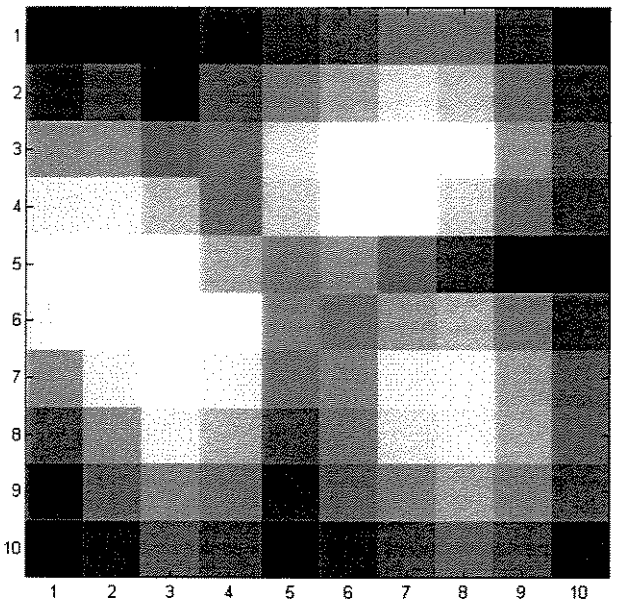


Figura 6.27 - Imagem da ativação média para o SOM apresentado na figura 3.20.

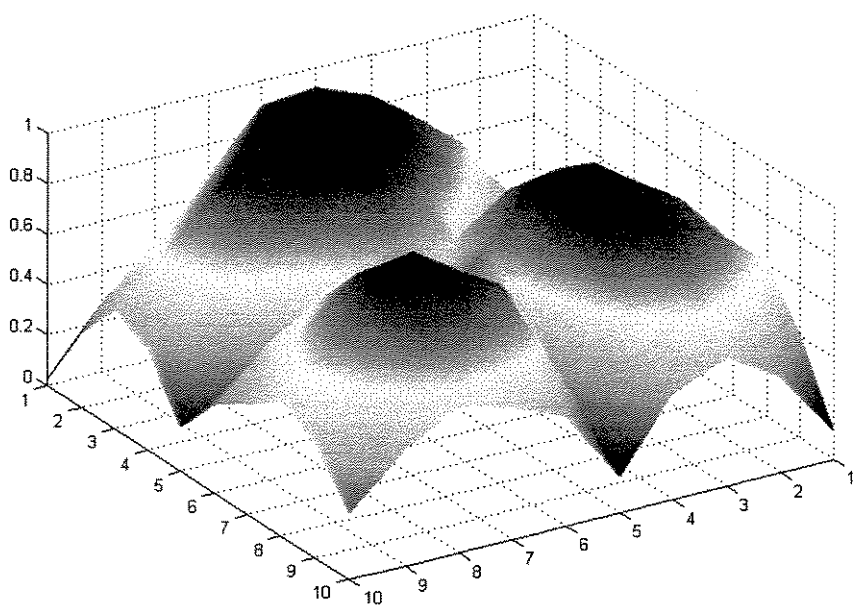


Figura 6.28 - Ativação média para o SOM apresentado na figura 3.20 como uma superfície.

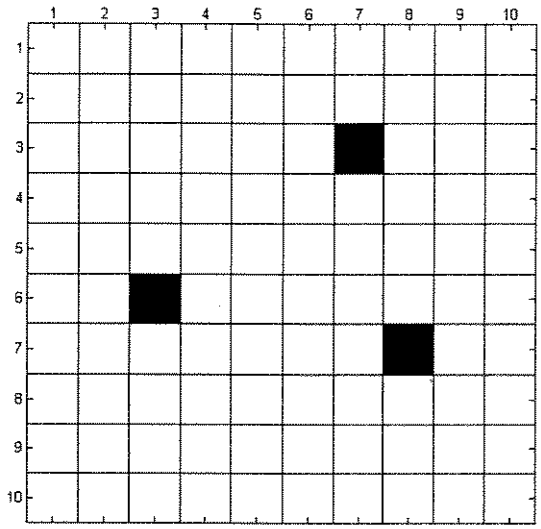


Figura 6.29 - Centros de ativação para o SOM apresentado na figura 3.20.

É interessante que usando os centros de ativação, como mostrado na figura 6.29, como marcadores da *watershed* para a *U-matrix*, para este problema (configuração do mapa apresentada na figura 3.20 e *U-matrix* apresentada nas figuras 5.3 e 5.4) o resultado da partição do mapa em regiões pela *watershed* foi idêntico ao resultado obtido pelos marcadores detectados pelo SL-SOM considerando o método apresentado na seção 5.3.2, que usa o gráfico das regiões conectadas *versus* o limiar de *U-matrix* (marcadores mostrados na figura 5.7). A figura 6.30 ilustra as linhas de *watershed* sobrepostas à *U-matrix* quando usamos os centros de ativação, mostrados na figura 6.29, como marcadores. Note que tal figura é igual à figura 5.9. Comparando numericamente as duas imagens, i.e., subtraindo a imagem apresentada na figura 6.30 da figura 5.9, foi obtido uma imagem nula, i.e., todos os elementos nulos, o que implica em igualdade do resultado do *watershed* pelos dois métodos de escolha de marcadores diferentes. O uso de centros de ativação como marcadores para a *watershed*, além de apresentarem a informação bastante clara do número de agrupamentos ou regiões no mapa, são pontuais, i.e., cada marcador é formado apenas por um pixel.

Deve-se notar que há diferença na informação dos marcadores obtidos pelo método apresentado na seção 5.3.2 dos marcadores obtidos pelos centros de ativação. Nestes últimos, todo conjunto de dados X é usado para gerar uma imagem de ativações médias no mapa, enquanto que no primeiro, a configuração estática do mapa após o treinamento dá origem à *U-matrix*, a qual é suavizada e posteriormente limiarizada em vários níveis a fim de que se possa detectar regiões de estabilidade em vales da superfície topográfica, que são candidatos a regiões (agrupamentos) do mapa.

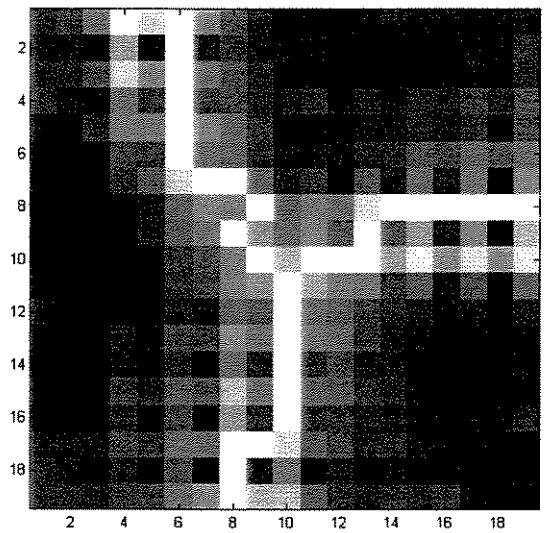


Figura 6.30: Linhas de watershed sobrepostas à U-matrix original, figura 5.3, em 2D, obtidas pelos marcadores encontrados pelos centros de ativação do mapa (figura 6.29).

6.3.4 Sumário das condições de parada da árvore do TS-SL-SOM

Sumarizando, para considerarmos a hipótese de que há mais de um agrupamento separável de dados em um dado SOM, deveríamos considerar a ocorrência, *concomitantemente*, das seguintes informações.

1. Os mapas deveriam apresentar mais de um centro de ativação. Idealmente o número de centros de ativação deveria ser igual ao número de agrupamentos nos dados, de forma a haver partição ótima dos dados pelo SOM, como ocorreu no problema ilustrado pela figura 6.30. Porém quando um agrupamento ocupa espaço ao redor de outro agrupamento ocorre uma composição dos estímulos e percebe-se apenas um grande agrupamento. O uso do gráfico das regiões conectadas é essencial para detectar este problema.

2. Histogramas de vencedores deveriam apresentar grande variabilidade (por exemplo, podemos checar o coeficiente de variabilidade), indicando presença de regiões de neurônios com elevada concentração de dados, separadas por regiões de baixa densidade de pontos. Estas regiões de baixa densidade de pontos é devida à característica do SOM em tentar representar a topologia do espaço de entrada, ao

mesmo tempo em que busca quantizar o espaço alocando mais neurônios nas regiões de maior concentração de dados. Pode-se fazer uma busca por regiões conectadas no histograma de vencedores da mesma forma que foi efetuada para a *U-matrix*, na seção 5.3.2, por exemplo, invertendo-se o histograma, $H_{novo} = 1 - H$, onde H foi escalonado linearmente no intervalo [0, 1].

3. Idealmente para haver agrupamentos temos que ter vales representativos na *U-matrix*, que são conseqüência das distâncias relativamente pequenas entre os neurônios que fazem parte de uma região que irá representar o agrupamento de dados. A maior parte dos pixels da imagem da *U-matrix* deveriam estar em valores relativamente baixos, enquanto que os pixels que representam bordas, de maior valor, deveriam ocorrer, com menor freqüência. Note, por exemplo nas figuras 5.3 e 5.22, que a maior parte dos pixels podem ser considerados como pertencentes a vales da *U-matrix*, e o processo de segmentação via *watershed* busca exatamente maximizar a separação entre pixels de bordas e vales (ver figuras 5.10 e 5.24). Note, na figura 6.5, onde temos um SOM treinado a partir de uma distribuição uniforme, que a maioria dos pixels não está em vales. No caso do SOM treinado com distribuição Gaussiana, a maioria dos pixels pode ser considerada como pertencente a vales, mas há apenas um vale significativo na *U-matrix*, o que implica em apenas um agrupamento dos dados (ver figuras 6.10, 6.12 e 6.14). Uma maneira relativamente simples para constatar a proporção de pixels em vales é considerar o histograma da imagem *U-matrix*, H_U. A figura 6.31 ilustra o histograma H_U para a *U-matrix* apresentada na figura 6.5, que é o caso em que usamos uma distribuição uniforme para treinar o SOM. O número de *bins* (ou faixas) nos histogramas apresentados foi 60, em todos os casos, a menos que se explicite o contrário.

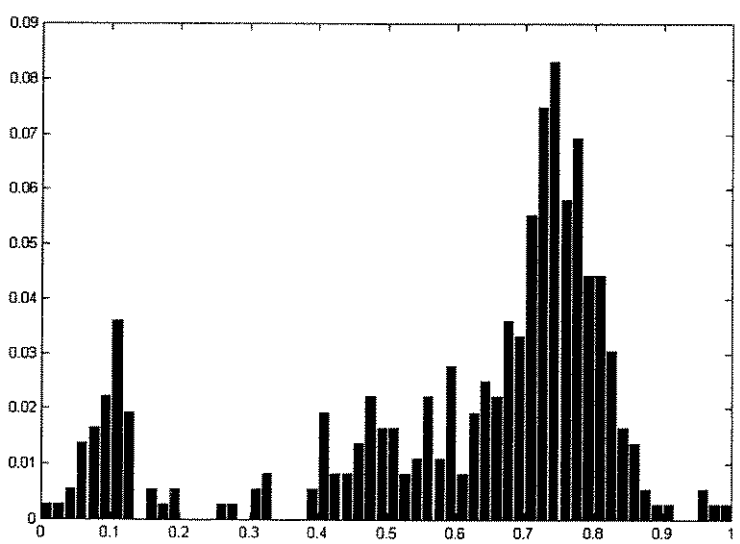


Figura 6.31 - Histograma para a U-matrix apresentada na figura 6.5.

Note que grande parte dos pixels não estão sendo alocados para vales da *U-matrix*, o que pode ser visto pelo deslocamento à direita do histograma. No caso da distribuição uniforme, o histograma da *U-matrix* está deslocado para a esquerda, devido à grande concentração de neurônios no centro da distribuição (figura 6.32). Porém, note que estamos usando apenas a informação de nível de cinza dos pixels, que está relacionada às distâncias entre os neurônios. Não há noção de conectividade nestes histogramas, e como foi discutido anteriormente, esta concentração do histograma à esquerda não implica, sozinha, na presença de mais de um agrupamento. De fato, neste caso temos apenas um grande vale, como mostrado na figura 6.14.

Para o problema apresentado na figura 3.20, cuja *U-matrix* é mostrada na figura 5.3, temos o histograma H_U mostrado na figura 6.33. Note que, de forma semelhante à figura 6.32, grande parte dos pixels possuem valores baixos, i.e., estão em regiões de vale da *U-matrix*.

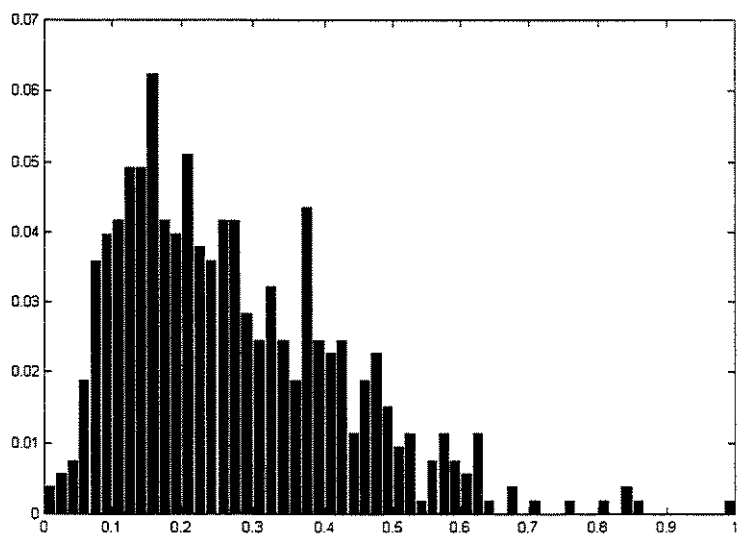


Figura 6.32 - Histograma para a U-matrix apresentada na figura 6.12.

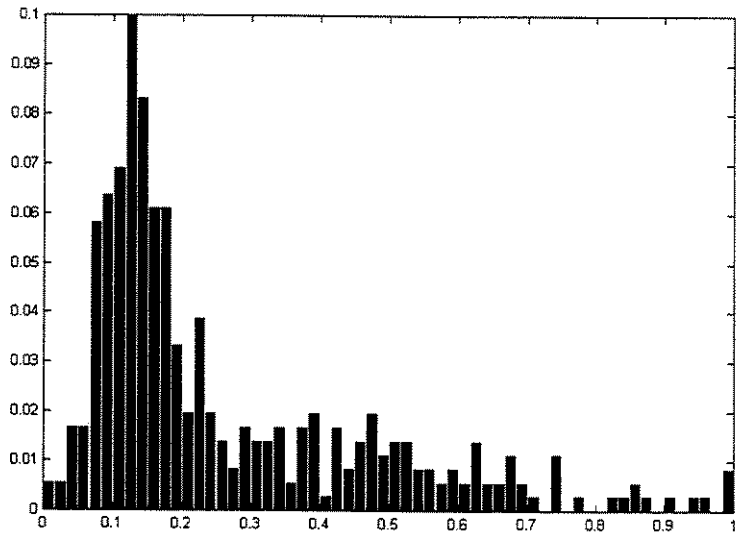


Figura 6.33 - Histograma para a U-matrix apresentada na figura 5.3.

Existem várias maneiras de detectar automaticamente o deslocamento à esquerda do histograma H_U. Podemos, por exemplo, dividir o histograma em regiões e computar a freqüência relativa dos valores dos pixels. Por exemplo, para o histograma apresentado na figura 6.33, caso dividamos em quatro regiões de mesmo tamanho, nos intervalos [0, 0.25], [0.25, 0.50], [0.50, 0.75] e [0.75, 1.0], obtemos o gráfico apresentado na figura 6.34. Na realidade, este gráfico é igualmente um histograma H_U porém com apenas 4 *bins*.

No primeiro *bin* temos 65.10% de todos os pixels da *U-matrix*. No segundo temos 19.67%, no terceiro 11.91% e no quarto 3.32%. Esta informação é pertinente quando busca-se agrupamentos na *U-matrix* pois como ressaltado anteriormente, a maioria dos pixels deveria ter nível de cinza relativamente baixos, o que caracteriza concentrações de neurônios para representar agrupamentos de dados.

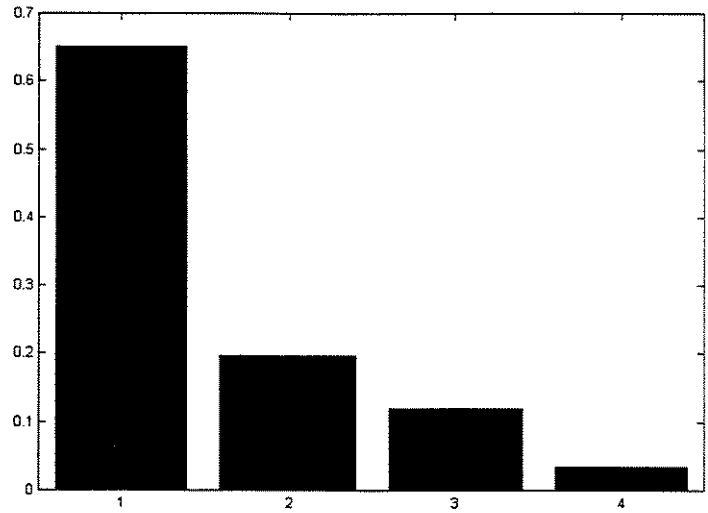


Figura 6.34 - Histograma para a U-matrix apresentada na figura 5.3 usando apenas 4 bins .

Outra forma de detectar a informação automaticamente seria fazer um histograma cumulativo a partir de H_U, i.e., $H_U{}^C$. Neste caso, o valor $y = f(x)$ para cada x é o somatório de todos os *bins* de zero a x, i.e., a integral de 0 a x do histograma da *U-matrix* H_U.

O histograma cumulativo para a figura 6.33 é apresentada na figura 6.35. Note que há um rápido crescimento no início do gráfico, $f(x) > x$, para valores relativamente baixos de x, e à medida que x cresce, a derivada da curva aproxima-se de zero. O histograma cumulativo para as figuras 6.31 e 6.32 são mostrados nas figuras 6.36 e 6.37, respectivamente.

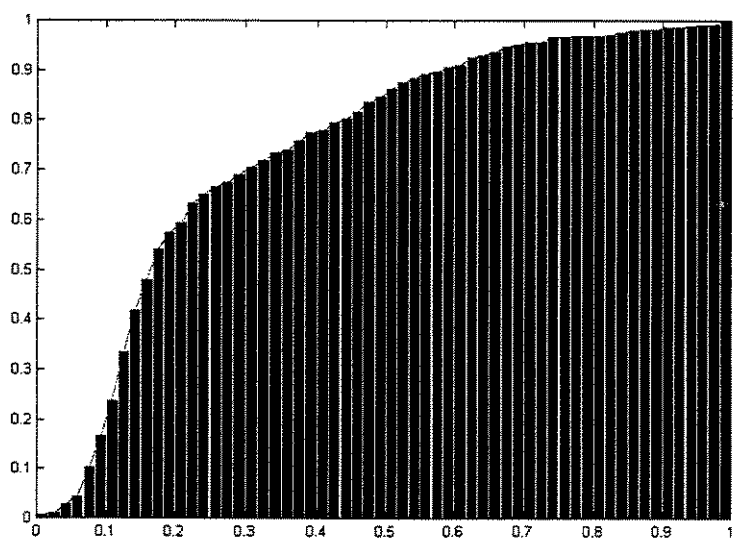


Figura 6.35 - Histograma cumulativo para a figura 6.33.

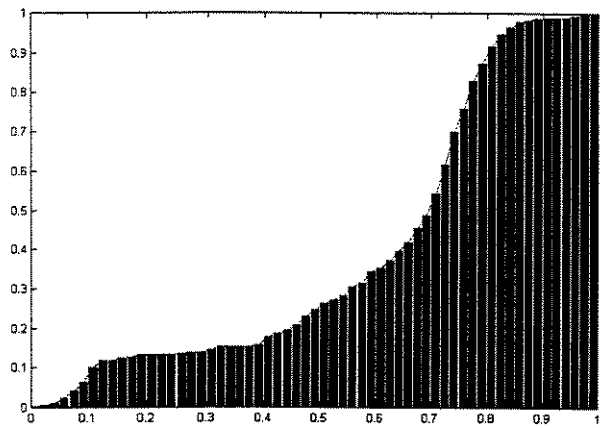


Figura 6.36 - Histograma cumulativo para a fig. 6.31

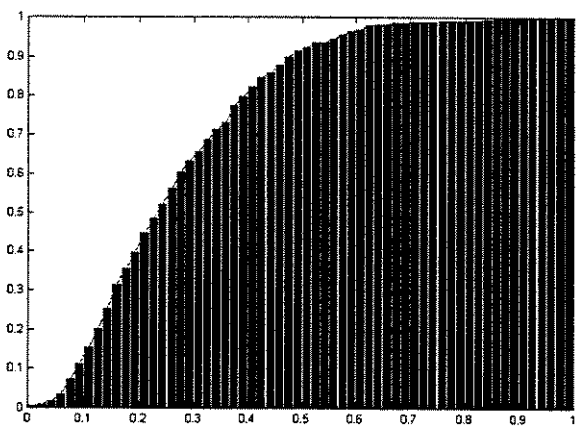


Figura 6.37 - Histograma cumulativo para a fig. 6.32

A automação da obtenção da informação de derivada elevada no início e baixa no final de $f(x)$, que representa o $H_U{}^C$, na faixa de valores de x, pode ser feita de forma relativamente simples. Podemos dividir o histograma cumulativo em faixas, por exemplo, três, e em cada faixa efetuar uma regressão linear. Teremos três polinômios de grau 1 onde os valores dos coeficientes podem nos informar a respeito do comportamento do histograma.

Para o caso da figura 6.35, temos os polinômios de grau 1, $f(x) = a_1 \cdot x + a_0$, dados pelos coeficientes $(a_1, a_0) = (2.641, -0.027)$, $(0.667, 0.511)$ e $(0.145, 0.854)$ para $f_1(x)$, $f_2(x)$ e $f_3(x)$, respectivamente, onde $f_1(x)$ foi obtida pela regressão linear na faixa de valores de x compreendendo o intervalo [0.0, 0.33]. Os outros dois polinômios $f_2(x)$ e $f_3(x)$ foram obtidos usando os intervalos seguintes de x, i.e., [0.33, 0.66] e [0.66, 1.0], respectivamente.

Note que, no caso ideal, em que deve haver agrupamentos, $a_1(f_1) > a_1(f_2) > a_1(f_3)$ e $a_0(f_1) < a_0(f_2) < a_0(f_3)$. Note que esta é uma das condições, não a única, pois o caso da distribuição Gaussiana satisfaz (veja a figura 6.36), porém esta figura foi obtida de um SOM treinado apenas com uma população ou agrupamento de dados.

4. Quando desconsideramos os efeitos dos neurônios em posições de borda do mapa o sistema deve ser capaz de identificar regiões conectadas estáveis significativas, o que não ocorre nas figuras 6.7 e 6.16, que são os casos das distribuições uniforme e Gaussiana, respectivamente. Podemos estabelecer um valor limiar o qual previne contra aparecimentos de picos ou pequenas oscilações como ocorreu na figura 6.16. Consideraremos que o limiar de número de regiões conectadas diferentes de 1 seja definido por $\gamma = (\text{NRC} \neq 1) / (\text{NRC} = 1)$, onde $\text{NRC} \neq 1$ significa o número de níveis de cinza da *U-matrix* com mais de uma região conectada. Assumiremos neste trabalho, a

menos que se explicite o contrário que γ_{min} seja 20%. Caso uma *U-matrix* não atinja esta meta consideraremos que não haverá segmentação do mapa.

5. Ainda existem outras formas de extrair informações a respeito da estrutura do SOM treinado, como discutimos nos capítulos 3 e 5 a respeito dos neurônios inativos. Eliminando a influência dos neurônios inativos pela cópia do vetor do neurônio ativo mais próximo, temos uma *U-matrix* que apresenta maior descontinuidades entre os agrupamentos de neurônios, o que reflete na descoberta do número de agrupamentos mais próximo do real.

6.4. Exemplos de aplicação em conjuntos de dados

Esta seção apresenta alguns resultados do TS-SL-SOM para alguns conjuntos de dados. Comentários sobre os resultados de cada conjunto de dados ajudam a ilustrar o funcionamento do método.

6.4.1 Conjunto de dados gerado artificialmente

O conjunto de dados abaixo, apresentado na figura 6.38, foi gerado artificialmente com 1876 padrões, onde as classes 1 a 5 possuem, respectivamente, 157, 122, 772, 642 e 183 padrões.

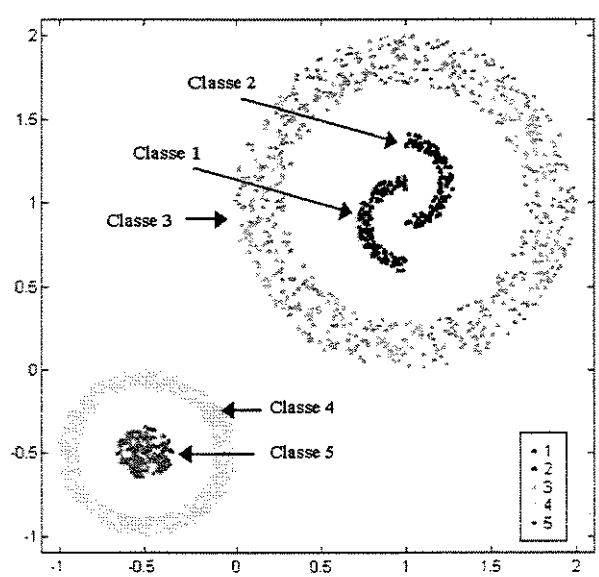


Figura 6.38: Conjunto de dados gerado artificialmente

Para o mapa raiz do TS-SL-SOM foi usado um SOM com topologia retangular com tamanho 15×15. A vizinhança inicial e final teve raios 12 e 1, respectivamente. O raio de vizinhança inicial de todos os mapas é calculada como sendo 80% da dimensão do mapa, arredondando para o inteiro mais próximo. O algoritmo usado foi o batch, e o número máximo de iterações foi 500. O tempo de treinamento em um computador Pentium 166 MHz, rodando sob o Matlab foi aproximadamente 29 minutos. O erro de quantização alcançado foi 0.058364. A configuração de neurônios após treinamento é mostrada, juntamente com os dados, na figura 6.39.

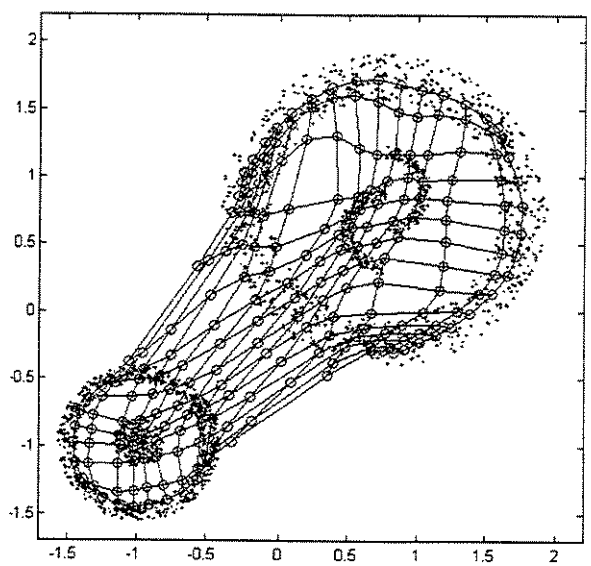


Figura 6.39: Dados e a configuração de neurônios após 500 iterações do algoritmo batch.

A *U-matrix* é apresentada na forma planar, em níveis de cinza na figura 6.40, e na forma de superfície na figura 6.41. O histograma de vencedores é apresentado na figura 6.42. Note na figura 6.43 o mapeamento efetuado pelo SOM, usando a informação privilegiada das classes, o que assumimos não conhecer.

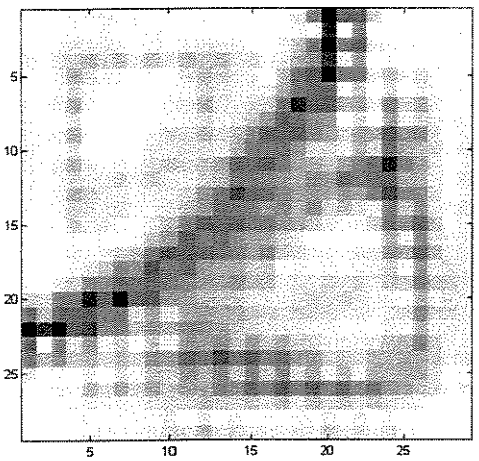


Figura 6.40: U-matrix – 2D

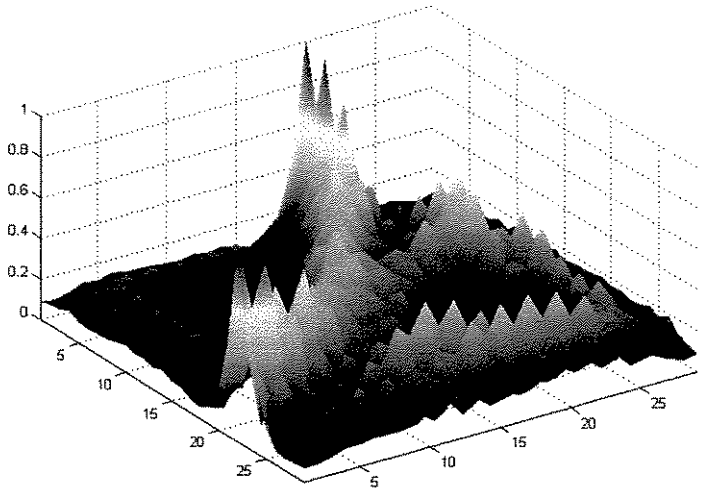


Figura 6.41: U-matrix – 3D

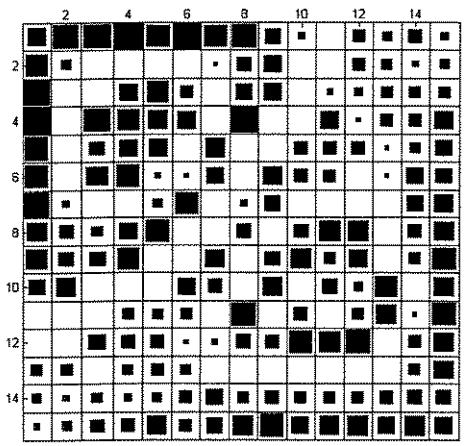


Figura 6.42: Histograma de vencedores

4	4	4	4	4	4	4	4	4	4		3	3	3	3
4	4					4	4	4			3	3	3	3
4			5	5	5		4	4		3	3	3	3	3
4		5	5	5	5		4			3	3	3	3	3
4		5	5	5		4			3	3	3	3	3	3
4		5	5	5	4	4		3	3	3		3	3	3
4	4			4	4		3	3					3	3
4	4	4	4	4			3		1	1	1		3	3
4	4	4	4			3		1	1	1	1		3	3
4	4				3	3		1		1	1	2		3
				3	3	3		1		2		2	2	3	3
			3	3	3	3	1	1	2	2	2	2		3	3
3	3			3	3	3								3	3
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

Figura 6.43: Mapeamento das classes no SOM

As figuras 6.44 e 6.45 ilustram a ativação média do mapa, onde podemos perceber dois centros de ativação, localizados nas posições do mapa (5, 5) e (10, 9).

O gráfico do número de regiões conectadas *versus* o limiar da *U-matrix* é apresentado na figura 6.46. Note que o patamar mais estável ocorre para 2 regiões, no intervalo dos níveis de cinza de 75 a 126. Desta forma, estamos separando o mapa em duas grandes regiões, correspondendo uma ao grupo correspondente das classes 1 a 3, e o outro correspondente às classes 4 e 5.

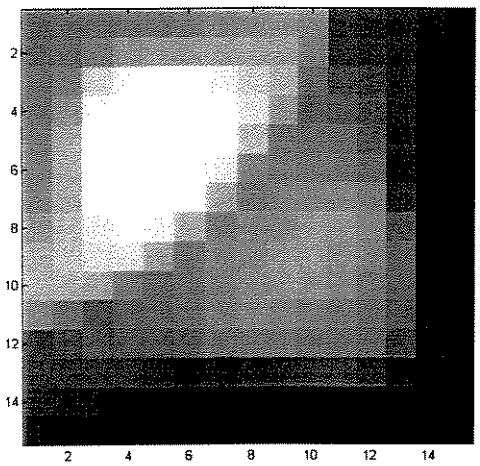


Figura 6.44: Ativação do mapa – 2D

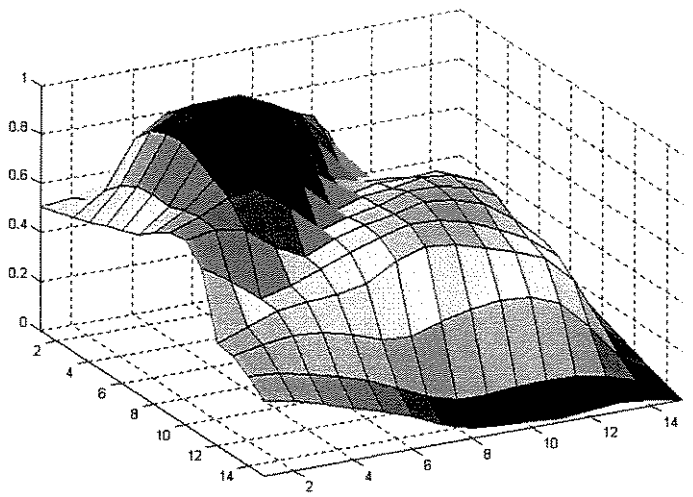


Figura 6.45: Ativação do mapa – 3D

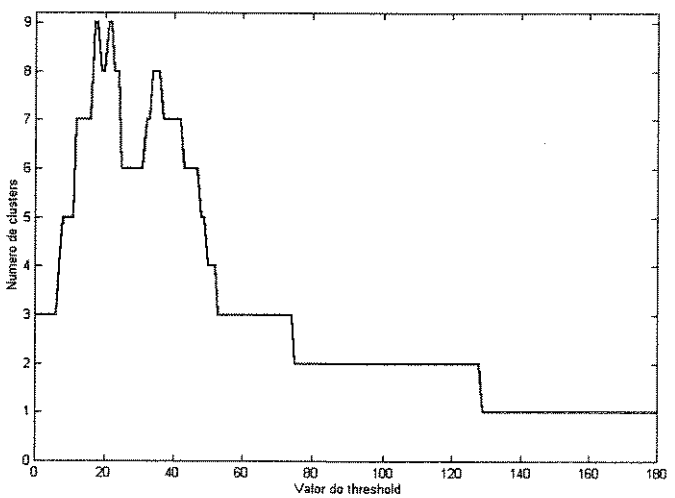


Figura 6.46: Gráfico do número de regiões conectadas versus o limiar da U-matrix

A figura 6.47 ilustra o histograma da *U-matrix* após escalonamento linear dos níveis de cinza. Note que tal histograma está deslocado para a esquerda, o que significa que a maioria dos pixels possui valores baixos, i.e., estão em regiões de vales da *U-matrix*. A imagem de marcadores (veja figura 6.48), como especificado pelo algoritmo SL-SOM, é a imagem resultante da limiarização da *U-matrix* pelo valor inicial (75) do platô de estabilidade de regiões conectadas (veja figura 6.46).

A figura 6.49 mostra a partição encontrada pelo algoritmo *watershed*. As figuras 6.50 e 6.51 ilustram, respectivamente, a sobreposição das linhas de *watershed* sobre a *U-matrix* em 2D e 3D. Após rotulagem da *U-matrix* segmentada (figura 6.52), os códigos das regiões

foram passados aos neurônios correspondentes. Apenas 3 neurônios não tiveram códigos diretamente da *U-matrix* (figura 6.53), e usamos o algoritmo vizinhos mais próximos para rotular tais neurônios (figura 6.54).

Note na figura 6.55 a partição efetuada no mapa. As duas grandes classes, a qual denominaremos classe C^1 e C^2 correspondem às classes de dados {4, 5} e {1, 2, 3}, respectivamente.

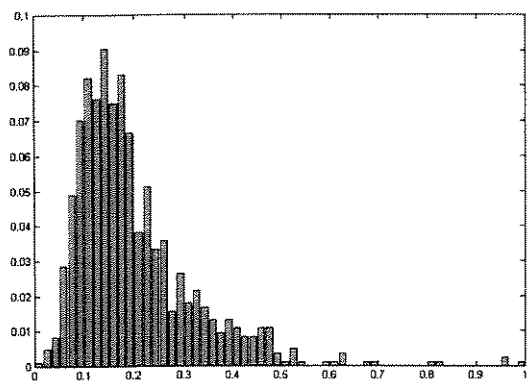


Figura 6.47 - Histograma da U-matrix

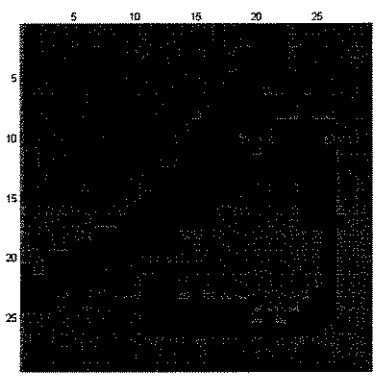


Figura 6.48 - Marcadores encontrados

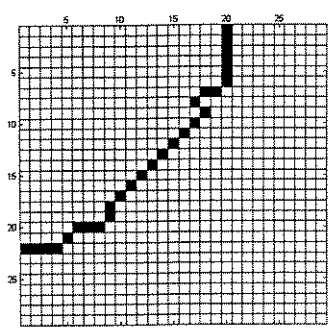


Figura 6.49 - Partição encontrada pelo algoritmo watershed

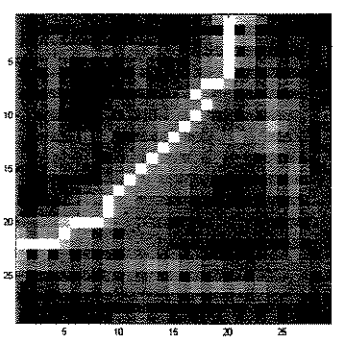


Figura 6.50 - Sobreposição das linhas de watershed sobre a U-matrix em 2D.

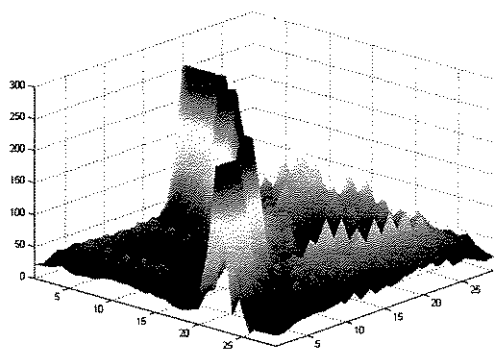


Figura 6.51 - Sobreposição das linhas de watershed sobre a U-matrix em 3D.

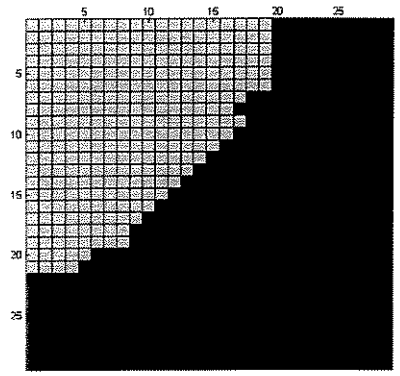


Figura 6.52 - U-matrix rotulada após segmentação pela watershed.

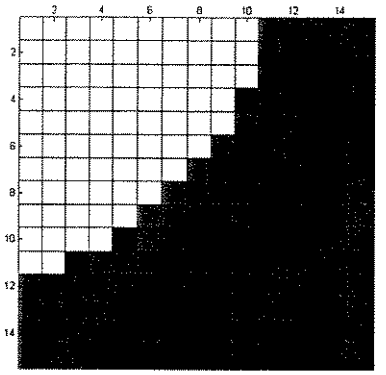


Figura 6.53 - SOM rotulado a partir dos códigos das regiões da U-matrix. Note que 3 neurônios não estão rotulados.

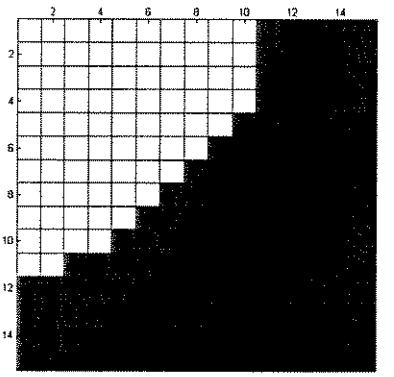


Figura 6.54 - SOM totalmente rotulado pelo uso do algoritmo vizinhos mais próximos para rotular os 3 neurônios não rotulados na figura 6.53.

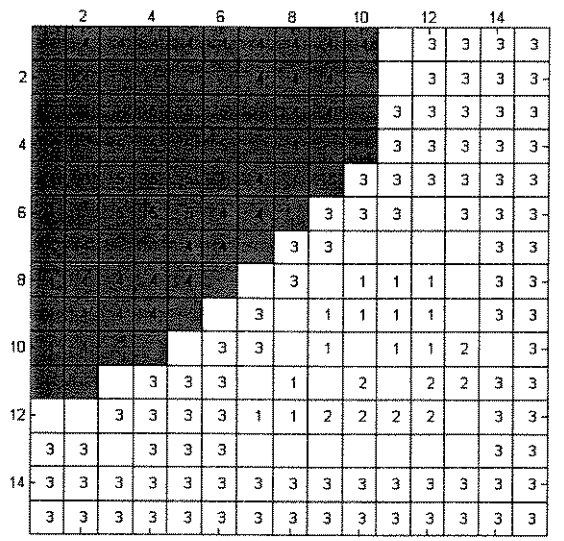


Figura 6.55 - SOM rotulado e o mapeamento das classes reais no mapa

Retirando o efeito dos neurônios inativos, i.e., todos os neurônios cujo $H(i, j) < 1$, onde (i, j) denota a posição do neurônio no mapa, temos os resultados apresentados nas figuras 6.56 a 6.59. Note a mudança da configuração de neurônios apresentada na figura 6.56 em relação à configuração original apresentada na figura 6.39. A U-matrix correspondente à figura 6.56 é apresentada na figura 6.57 (na forma bidimensional) e na figura 6.58 (na forma tridimensional).

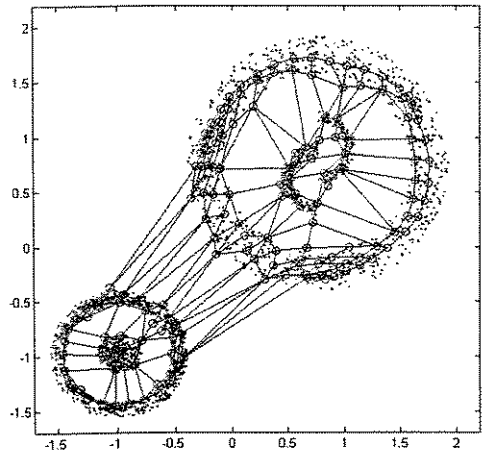


Figura 6.56: Dados e a configuração de neurônios eliminando o efeito dos neurônios inativos, $H(i, j) < 1$.

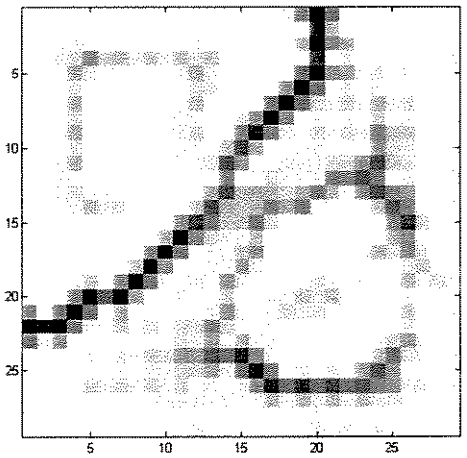


Figura 6.57: U-matrix (2D) correspondente à configuração de neurônios apresentada na figura 6.56.

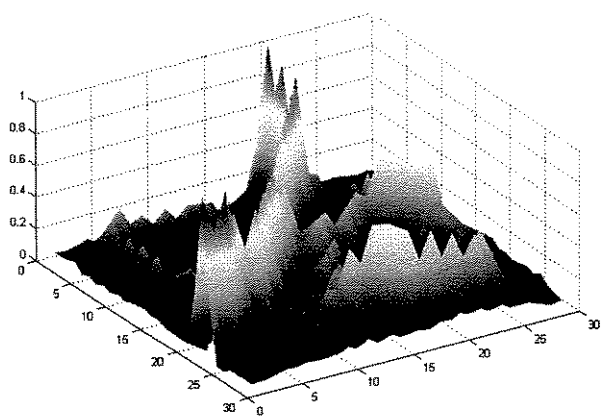


Figura 6.58: U-matrix (3D) correspondente à configuração de neurônios apresentada na figura 6.56.

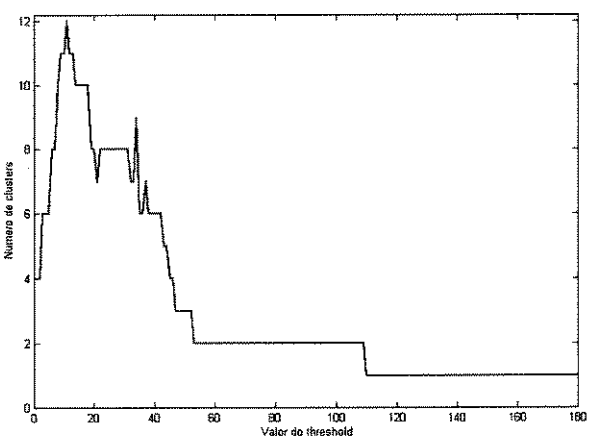


Figura 6.59: Gráfico do número de regiões conectadas para valores de limiares da U-matrix, para o SOM apresentado na figura 6.56.

Comparando a *U-matrix* do SOM convencional (figuras 6.40 e 6.41) com a *U-matrix* do SOM que teve o efeito dos neurônios inativos eliminado (figuras 6.57 e 6.58), vemos esta última apresenta uma melhor resolução dos agrupamentos. Note que o platô que indica dois agrupamentos é cerca de 10% maior na figura 6.59 do que o platô apresentado na figura 6.46. O intervalo para duas regiões conectadas estáveis está entre os valores de limiares 53 e 110.

Continuando o processo de geração dinâmica da árvore de mapas, temos o resultado final apresentado na figura 6.60. Iremos, a seguir, detalhar alguns resultados dos sub-mapas derivados do mapa raiz.

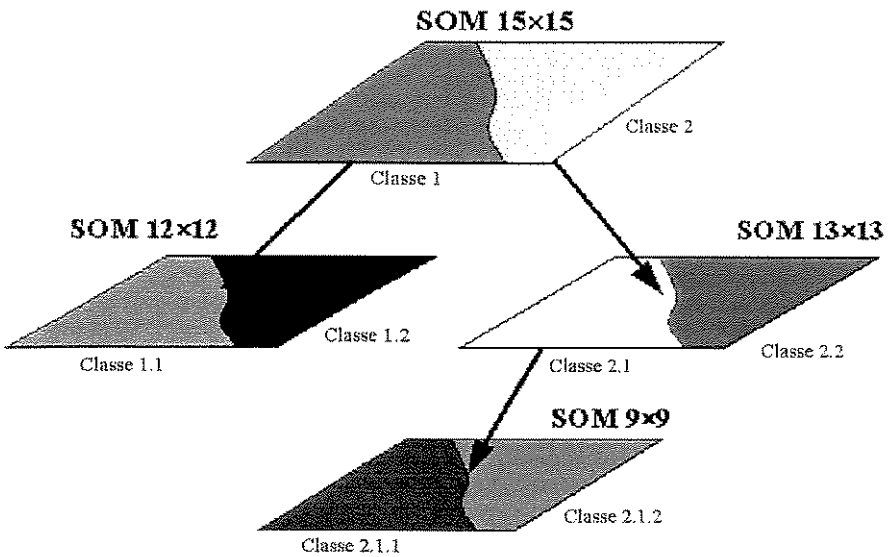


Figura 6.60 - Resultado final (ilustrativo) do TS-SL-SOM para o conjunto de dados apresentado na figura 6.38.

6.4.1.1 Sub-mapa 1

O sub-mapa 1 é filho da região 1 (classe C^1) do mapa raiz. 825 dos 1876 padrões foram classificados como pertencentes a esta classe. O tamanho do sub-mapa foi 12×12 e a topologia foi a mesma do mapa pai (mapa raiz). A vizinhança inicial foi 10 e o número máximo de iterações, no algoritmo *batch*, foi 500. O tempo de treinamento foi de aproximadamente 11 minutos (em um PC com Pentium 166 MHz) e o erro de quantização obtido foi 0.090379.

A configuração de neurônios é apresentada na figura 6.61. O histograma de vencedores é mostrado na figura 6.62 e a figura 6.63 ilustra o mapeamento efetuado pelo SOM considerando a informação das classes reais. Nota-se a região de baixa densidade de pontos olhando a figura 6.62. O histograma de vencedores é útil principalmente quando não podemos visualizar a configuração de neurônios, como ocorre quando a dimensão dos dados, p, é maior que 3. Note que este mapa funciona como uma focalização da atenção na região 1 do mapa raiz, o que possibilita a detecção de agrupamentos mais específicos.

As figuras 6.64 e 6.65 ilustram a *U-matrix*, respectivamente em 2 e 3-D. O gráfico de número de regiões conectadas versus limiar é apresentado na figura 6.66. A imagem de marcadores encontrados pelo algoritmo SL-SOM é mostrada na figura 6.67.

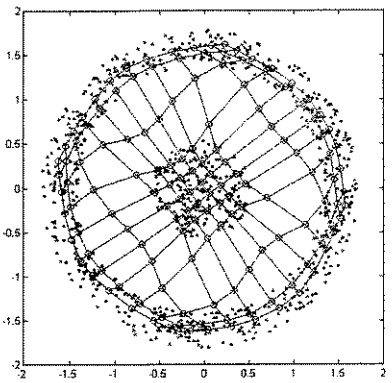


Figura 6.61 - Configuração de neurônios - sub-mapa 1.

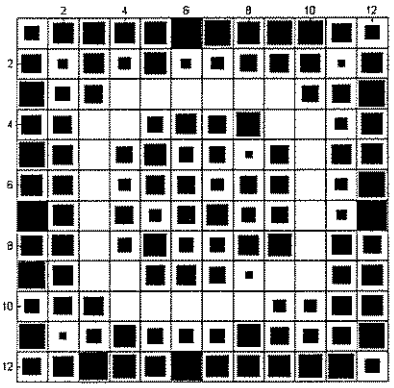


Figura 6.62 - Histograma de vencedores - sub-mapa 1.

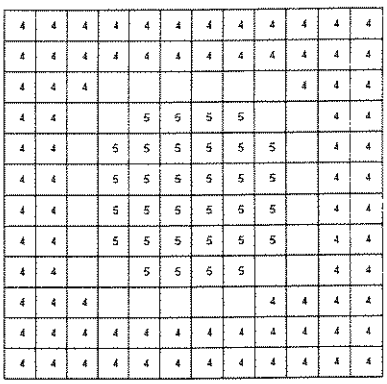


Figura 6.63 - Mapeamento das classes reais no sub-mapa 1.

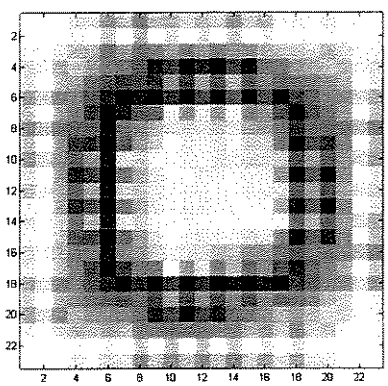


Figura 6.64 - U-matrix (2D) do sub-mapa 1.

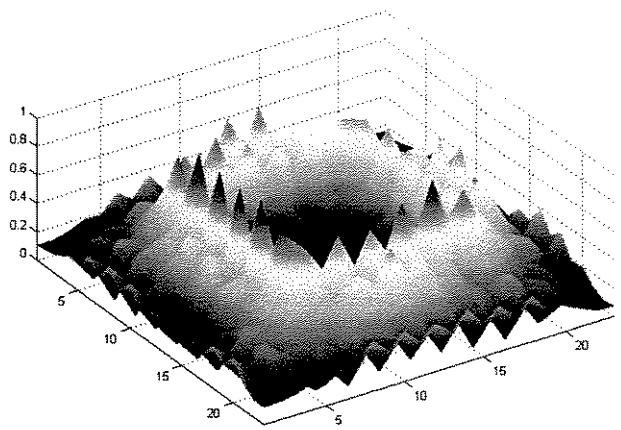


Figura 6.65 - U-matrix (3D) do sub-mapa 1.

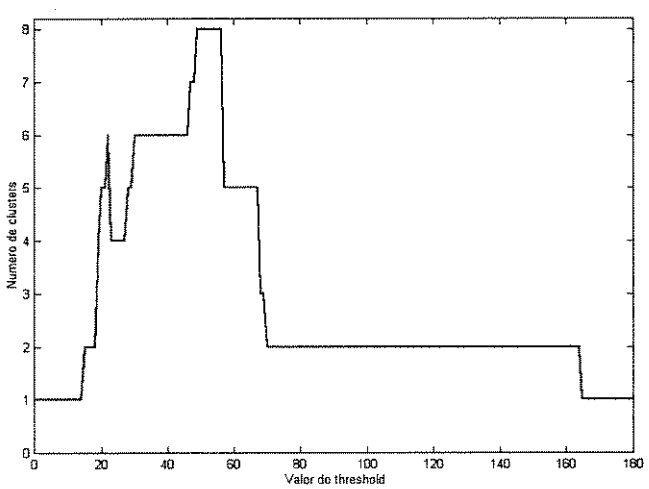


Figura 6.66 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 1

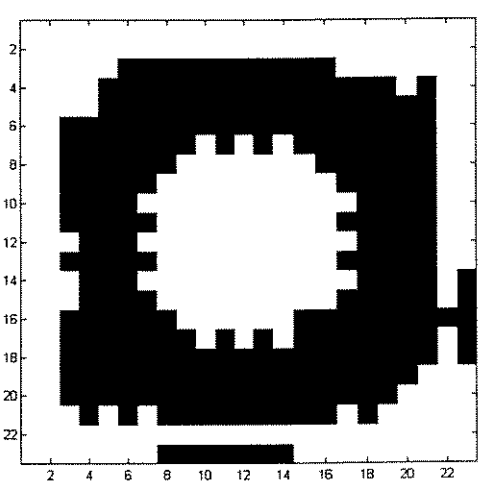


Figura 6.67 - Marcadores encontrados para a segmentação via watershed - sub-mapa 1

a-se na figura 6.66 uma estabilidade para o número de regiões conectadas (que é o nero de agrupamentos de neurônios) no valor 2, no intervalo de níveis de cinza 70 a 164. figuras 6.68 e 6.69 ilustram a ativação média do sub-mapa 1. Note que pelo fato de uma se estar dentro da outra a contribuição dos padrões se somam, e temos como resultado nas um centro de ativação. Assim, os centros de ativação não devem ser a única ormação para determinação do número de agrupamentos em um dado SOM, sendo de nde importância a análise a partir das regiões conectadas da *U-matrix*, como apresentado igura 6.66.

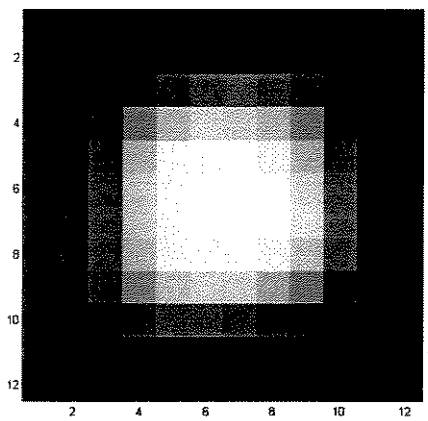


Figura 6.68 - Ativação média do sub-mapa 1 (em 2D).

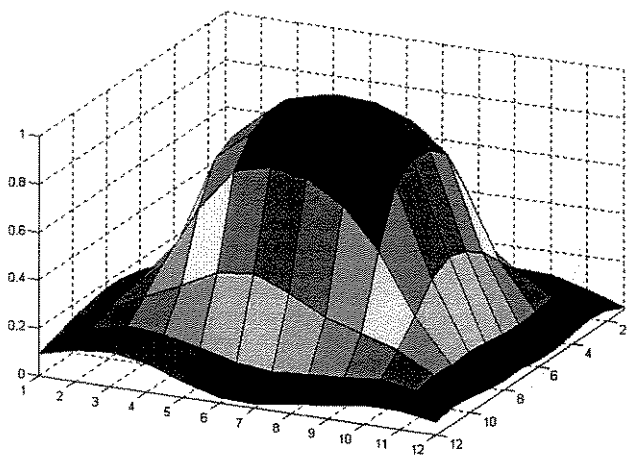


Figura 6.69 - Ativação média do sub-mapa 1 (em 3D)

ninando o efeito dos neurônios inativos do sub-mapa 1, temos a configuração de rônios apresentada na figura 6.70. Note que o gráfico de regiões conectadas (figura) para tal configuração é mais estável que o apresentado para a configuração original de rônios (figura 6.66).

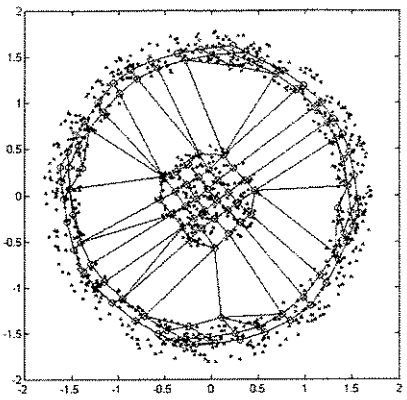


Figura 6.70 - *Dados e a configuração de neurônios (sub-mapa 1) eliminando o efeito dos neurônios inativos, $H(i, j) < 0$.*

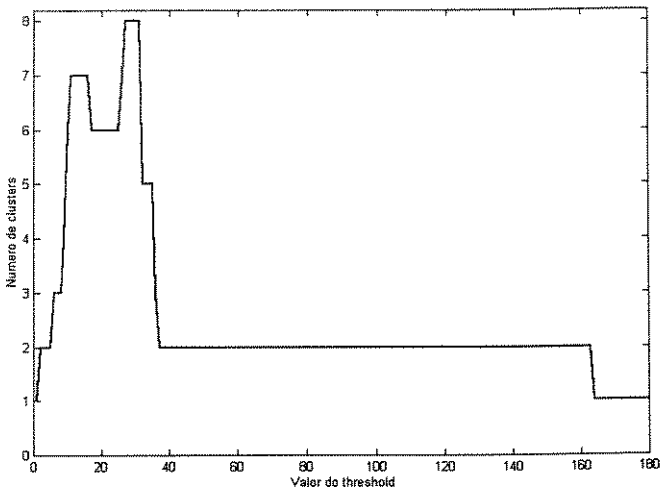


Figura 6.71 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 1- usando a configuração de neurônios apresentada na figura 6.70.

A *U-matrix* da configuração de neurônios apresentada na figura 6.70 é mostrada nas figuras 6.72 e 6.73. Note, comparando com as figuras 6.64 e 6.65 da *U-matrix* original, que as novas figuras apresentam maior separação entre os agrupamentos de neurônios. A estabilidade de regiões conectadas ocorre em um intervalo maior que o apresentado na figura 6.66, ocorrendo na faixa de níveis de cinza de 37 a 164. O platô de estabilidade, neste caso é cerca de 34% maior que o platô apresentado na figura 6.66. Note que usamos esta informação apenas como confirmatória do número de agrupamentos. O algoritmo de *watershed* é aplicado sempre à *U-matrix* original derivada do SOM obtido via treinamento.

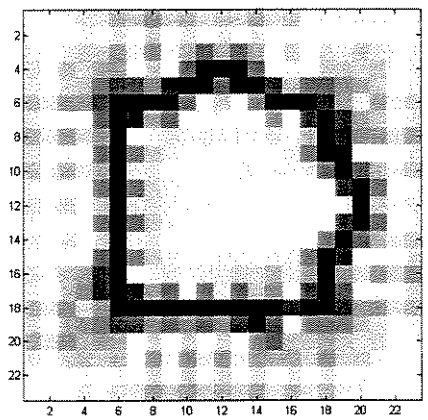


Figura 6.72 - U-matrix (2D) da configuração de neurônios apresentada na figura 6.70.

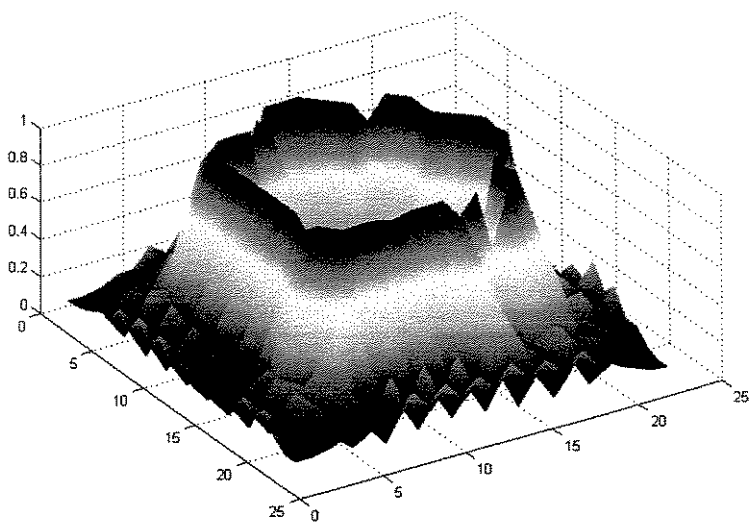


Figura 6.73 - U-matrix (3D) da configuração de neurônios apresentada na figura 6.70.

igura 6.74 ilustra a partição da *U-matrix* obtida após a aplicação do *watershed*. A figura 5 ilustra a sobreposição do contorno das regiões sobre a imagem da *U-matrix*.

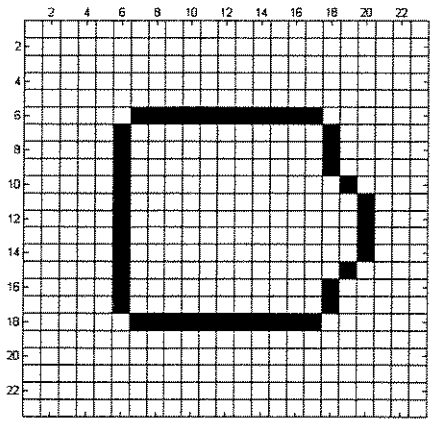


Figura 6.74 - Partição encontrada pelo algoritmo watershed (sub-mapa 1).

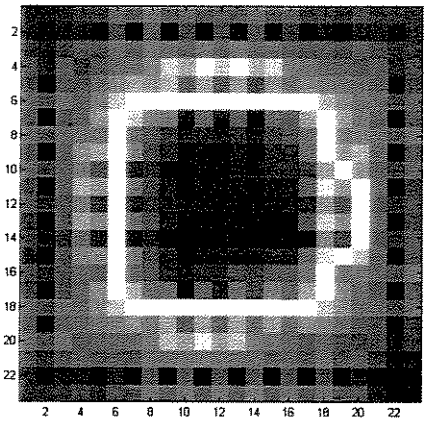


Figura 6.75 - Sobreposição das linhas de watershed sobre a imagem da U-matrix (sub-mapa 1).

igura 6.76 ilustra o sub-mapa 1 rotulado a partir dos códigos da *U-matrix* rotulada, e n a informação das classes reais mapeadas nos neurônios. A informação do mapeamento mesma que foi apresentada na figura 6.63, porém cada neurônio agora possui um rótulo, nível de cinza, de acordo com a pertinência aos dois agrupamentos encontrados. As ses C^{1-1} e C^{1-2} encontradas correspondem às classes reais dos dados 4 e 5 (veja a figura 3).

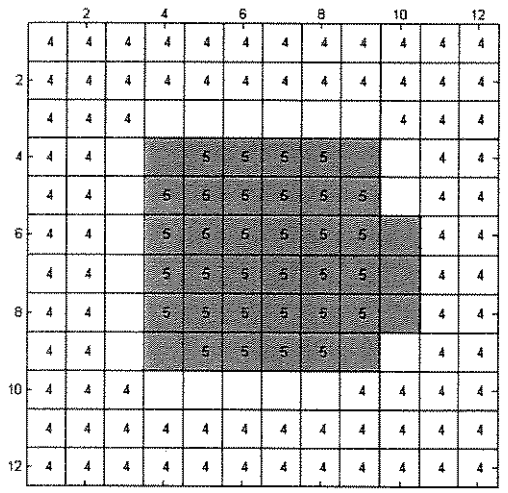


Figura 6.76 - Segmentação do sub-mapa 1, com as classes reais dos dados mapeadas nos neurônios

6.4.1.2 Sub-mapa 2

O sub-mapa 2 é filho da região 2 (classe C^2) do mapa raiz. 1051 dos 1876 padrões foram classificados como pertencentes a esta classe. O tamanho do sub-mapa foi 13×13 e a topologia foi a mesma do mapa pai (mapa raiz). A vizinhança inicial teve raio 10 e o número máximo de iterações, no algoritmo *batch*, foi 500. O tempo de treinamento foi de aproximadamente 16.3 minutos (em um PC com Pentium 166 MHz) e o erro de quantização obtido foi 0.099561.

A configuração de neurônios é apresentada na figura 6.77. O histograma de vencedores é mostrado na figura 6.78 e a figura 6.79 ilustra o mapeamento efetuado pelo SOM, considerando a informação das classes reais. A análise é similar à efetuada para o sub-mapa 1.

As figuras 6.80 e 6.81 ilustram a *U-matrix*, respectivamente em 2 e 3-D. O gráfico de número de regiões conectadas versus limiar é apresentado na figura 6.82. A imagem de marcadores encontrados pelo algoritmo SL-SOM é mostrada na figura 6.83.

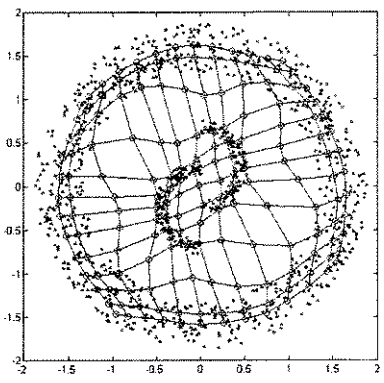


Figura 6.77 - Configuração de neurônios - sub-mapa 2.

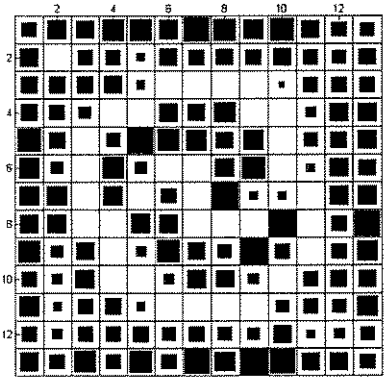


Figura 6.78 - Histograma de vencedores - sub-mapa 2.

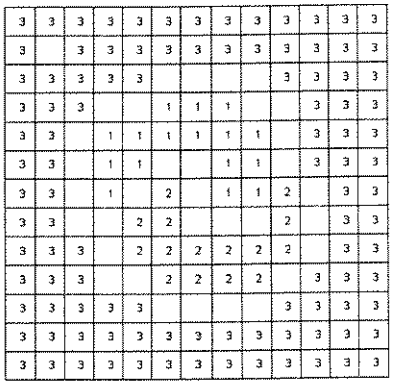


Figura 6.79 - Mapeamento das classes reais no sub-mapa 2.

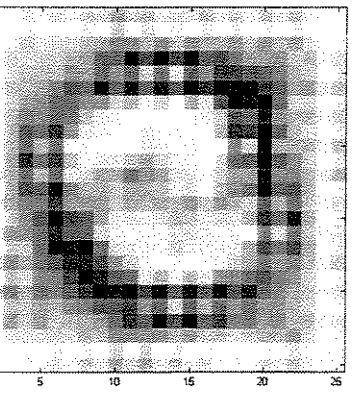


Figura 6.80 - U-matrix (2D) do sub-mapa 2.

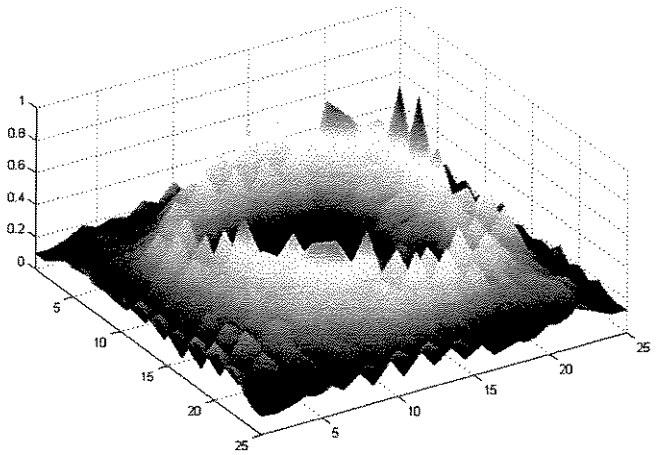


Figura 6.81 - U-matrix (3D) do sub-mapa 2.

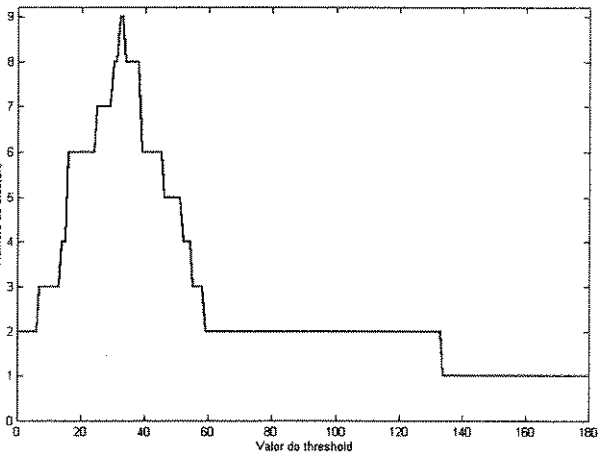


Figura 6.82 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 2

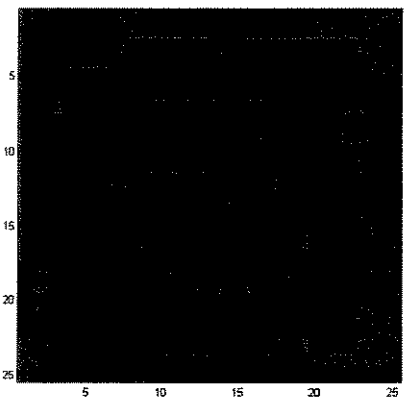


Figura 6.83 - Marcadores encontrados para a segmentação via watershed - sub-mapa 2

almente ao caso do sub-mapa 1, nota-se na figura 6.82 uma estabilidade para o número regiões conectadas no valor 2, no intervalo de níveis de cinza 59 a 133. As figuras 6.84 e 5 ilustram a ativação média do sub-mapa 2. Igualmente ao sub-mapa 1, as classes de os 1 e 2 estão dentro da região no espaço da classe 3 e as contribuições dos padrões se nam, resultando em apenas um centro de ativação.

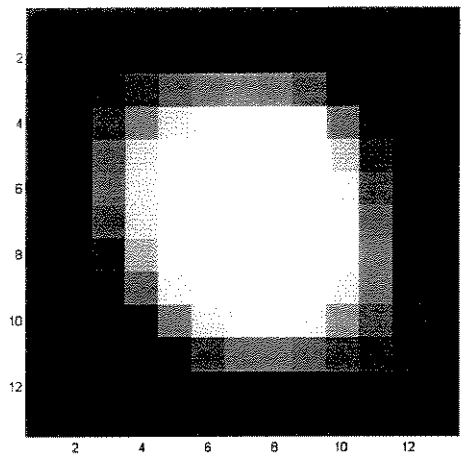


Figura 6.84 - Ativação média do sub-mapa 2 (em 2D).

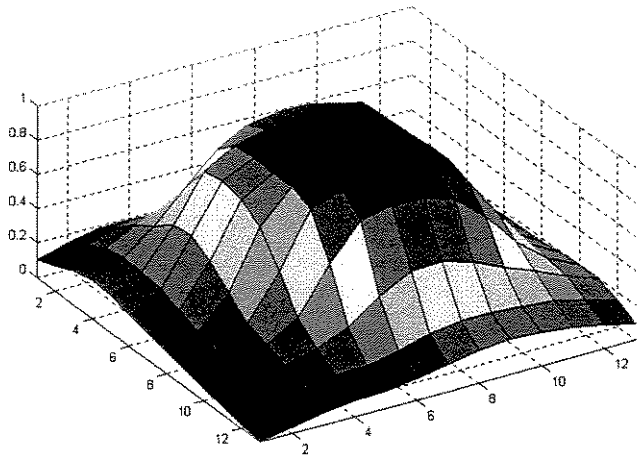


Figura 6.85 - Ativação média do sub-mapa 2 (em 3D)

Eliminando o efeito dos neurônios inativos do sub-mapa 2, temos a configuração de neurônios apresentada na figura 6.86. Note que o gráfico de regiões conectadas (figura 6.87) para tal configuração apresenta um platô de estabilidade também para 3 agrupamentos, sendo porém de menor tamanho que o platô para 2 agrupamentos.

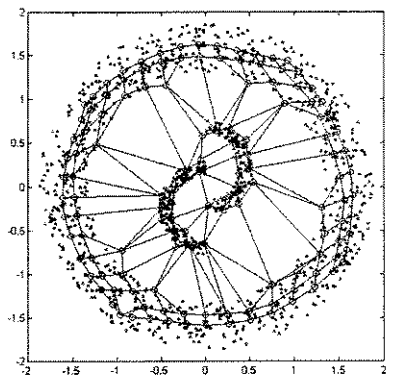


Figura 6.86 - *Dados e a configuração de neurônios (sub-mapa 2) eliminando o efeito dos neurônios inativos, $H(i, j) < 0$.*

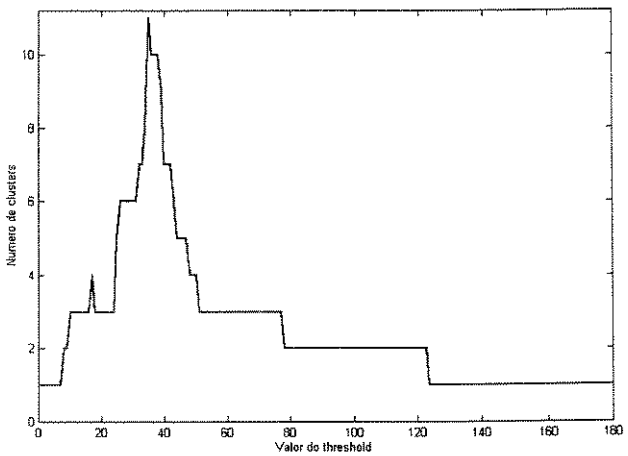


Figura 6.87 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 2- usando a configuração de neurônios apresentada na figura 6.86.

A *U-matrix* da configuração de neurônios apresentada na figura 6.86 é mostrada nas figuras 6.88 e 6.89. Note, comparando com as figuras 6.80 e 6.81 da *U-matrix* original, que as novas figuras apresentam maior separação entre os agrupamentos de neurônios. A figura

90 ilustra a partição da *U-matrix* obtida após a aplicação do *watershed*. A figura 6.91 stra a sobreposição do contorno das regiões sobre a imagem da *U-matrix*.

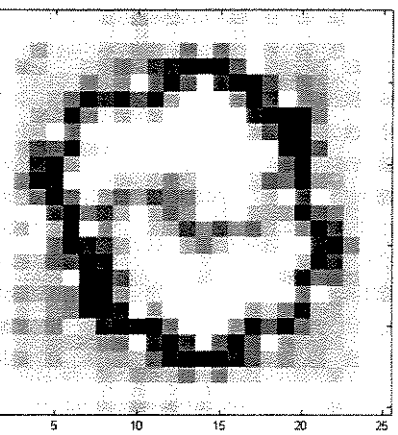


Figura 6.88 - U-matrix (2D) da configuração de neurônios apresentada na figura 6.86.

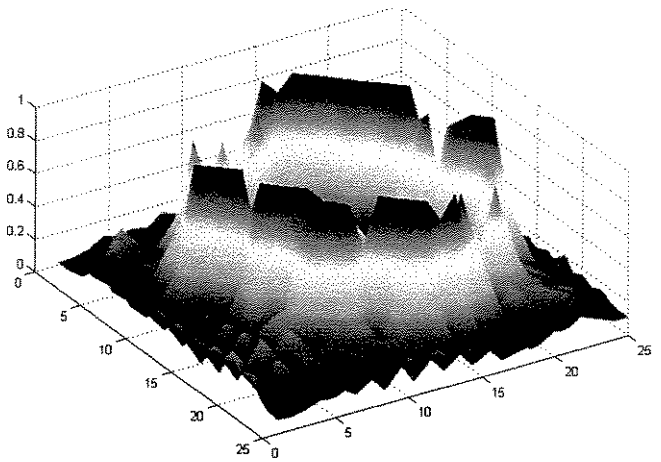


Figura 6.89 - U-matrix (3D) da configuração de neurônios apresentada na figura 6.86.

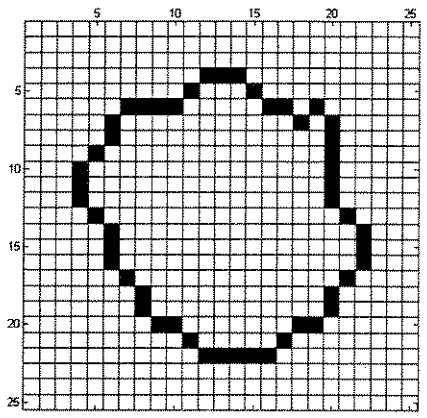


Figura 6.90 - Partição encontrada pelo algoritmo watershed (sub-mapa 2).

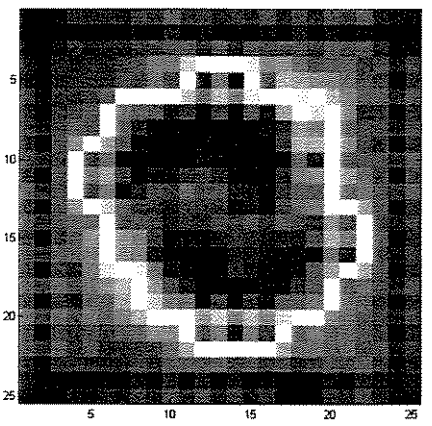


Figura 6.92 - Sobreposição das linhas de watershed sobre a imagem da U-matrix (sub-mapa 2).

igura 6.93 ilustra o sub-mapa 2 rotulado a partir dos códigos da *U-matrix* rotulada, e n a informação das classes reais mapeadas nos neurônios. A informação do mapeamento mesma que foi apresentada na figura 6.79, porém cada neurônio agora possui um rótulo, nível de cinza, de acordo com a pertinência aos dois agrupamentos encontrados. As sses C^{2-1} e C^{2-2} encontradas correspondem às classes reais dos dados {1, 2} e {3} (veja a ıra 6.38).

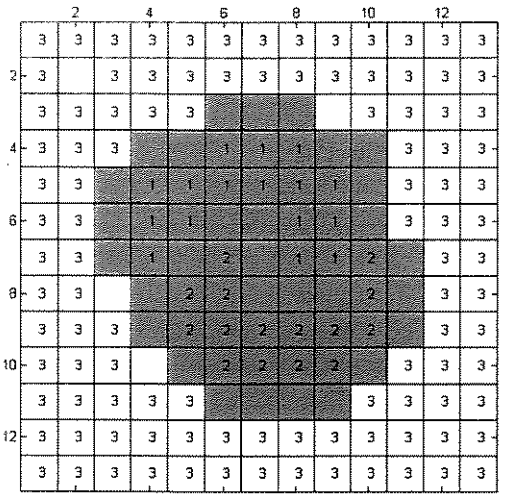


Figura 6.93 - Segmentação do sub-mapa 2 com as classes reais dos dados mapeadas nos neurônios

6.4.1.3 Sub-mapas 1-1, 1-2 e 2-2

Os sub-mapas 1-1, 1-2 e 2-2 foram criados porém como não houve detecção de mais de uma região foram eliminados, como descrito no algoritmo TS-SL-SOM. O sub-mapa 1-1, treinado com a classe 4 dos dados (ver figura 6.55), e o sub-mapa 2-2, treinado com a classe 3 dos dados (ver figura 6.93) apresentam configuração de neurônios e *U-matrix* similar ao exemplo mostrado nas figuras 5.73 a 5.78. O sub-mapa 1-2 foi treinado com distribuição uniforme e apresentou resultados semelhantes aos mostrados para as figuras 6.3 - 6.9. Considera-se que quando há eliminação de um sub-mapa em determinado nível da árvore, a região de neurônios que deu origem ao sub-mapa é suficiente para representar no espaço p-dimensional o subconjunto de dados utilizado.

6.4.1.4 Sub-mapa 2-1

O sub-mapa 2-1 é filho da região 1 (classe $C^{2\text{-}1}$) do sub-mapa 2. O tamanho do sub-mapa foi 9×9 e a topologia foi a mesma do mapa pai. A vizinhança inicial teve raio 7 e o número máximo de iterações, no algoritmo *batch*, foi 500. O tempo de treinamento foi de aproximadamente 2 minutos (em um PC com Pentium 166 MHz) e o erro de quantização obtido foi 0.110456.

A configuração de neurônios é apresentada na figura 6.94. O histograma de vencedores é mostrado na figura 6.95 e a figura 6.96 ilustra o mapeamento efetuado pelo SOM considerando a informação das classes reais. A análise é similar à efetuada para o sub-mapa

2. As figuras 6.97 e 6.98 ilustram a *U-matrix*, respectivamente em 2 e 3-D. O gráfico de número de regiões conectadas *versus* limiar é apresentado na figura 6.99. A imagem de marcadores encontrados pelo algoritmo SL-SOM é mostrada na figura 6.100.

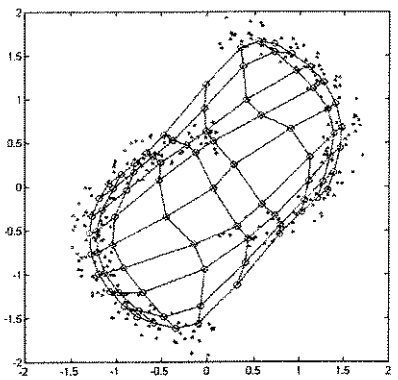


Figura 6.94 - Configuração de neurônios - sub-mapa 2-1.

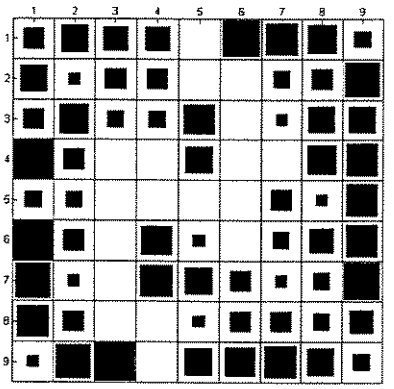


Figura 6.95 - Histograma de vencedores - sub-mapa 2-1.

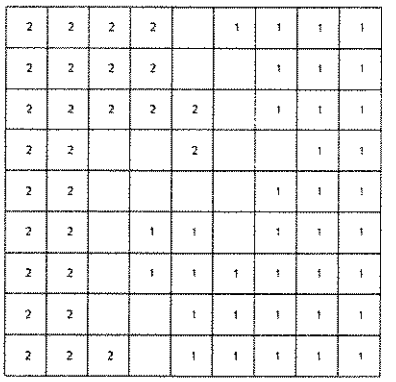


Figura 6.96 - Mapeamento das classes reais no sub-mapa 2-1.

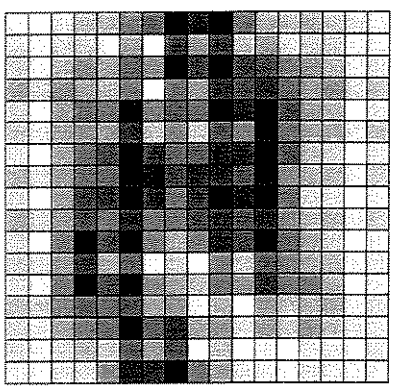


Figura 6.97 - U-matrix (2D) do sub-mapa 2-1.

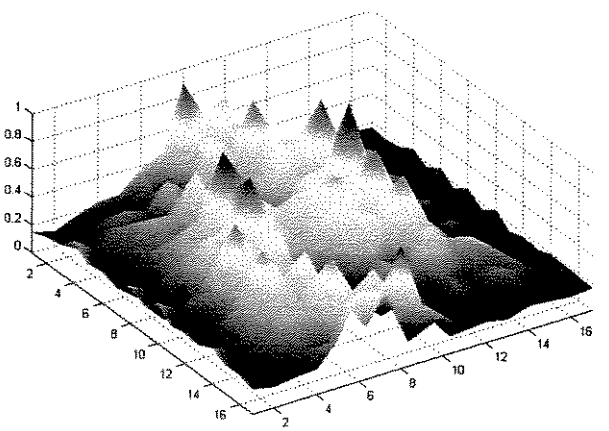


Figura 6.98 - U-matrix (3D) do sub-mapa 2-1.

Nota-se na figura 6.99 uma estabilidade para o número de regiões conectadas no valor 2, no intervalo de níveis de cinza 45 a 162. Eliminando o efeito dos neurônios inativos do sub-mapa 2-1, temos a configuração de neurônios apresentada na figura 6.101. A figura 6.102 ilustra o gráfico do número de regiões conectadas *versus* o nível de cinza da U-matrix da configuração de neurônios apresentada na figura 6.101. Note, igualmente ao caso da figura 6.99, um platô de estabilidade para 2 agrupamentos de neurônios.

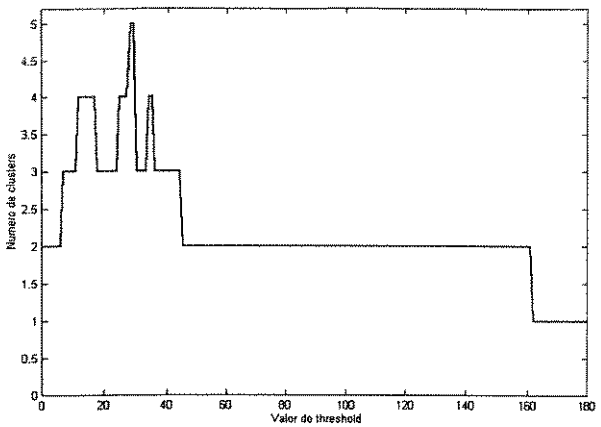


Figura 6.99 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 2-1.

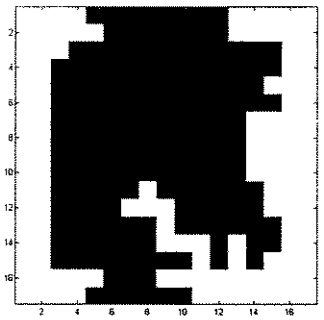


Figura 6.100 - Marcadores encontrados para a segmentação via watershed - sub-mapa 2-1

A *U-matrix* da configuração de neurônios apresentada na figura 6.101 é mostrada nas figuras 6.103 e 6.104. Note, comparando com as figuras 6.97 e 6.98 da *U-matrix* original, que as novas figuras apresentam maior separação entre os agrupamentos de neurônios. A figura 6.105 ilustra a partição da *U-matrix* obtida após a aplicação do *watershed*. A figura 6.106 ilustra a sobreposição do contorno das regiões sobre a imagem da *U-matrix*.

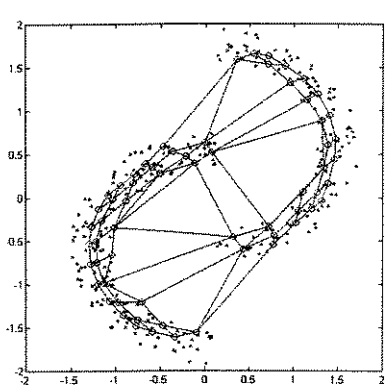


Figura 6.101 - *Dados e a configuração de neurônios (sub-mapa 2-1) eliminando o efeito dos neurônios inativos, $H(i, j) < 0$.*

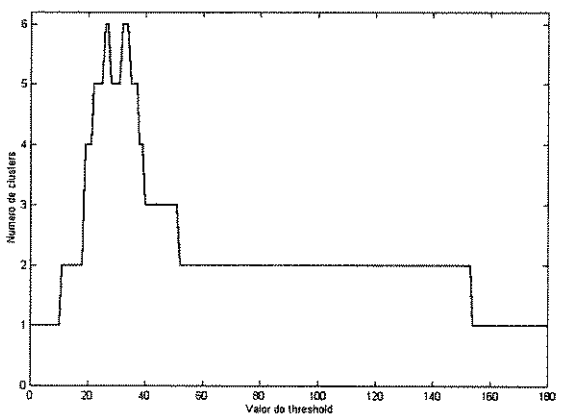


Figura 6.102 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 2-1- usando a configuração de neurônios apresentada na figura 6.101.

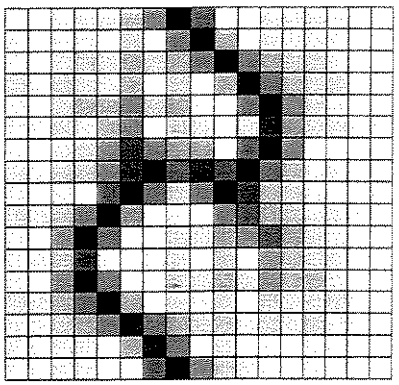


Figura 6.103 - U-matrix (2D) da configuração de neurônios apresentada na figura 6.101.

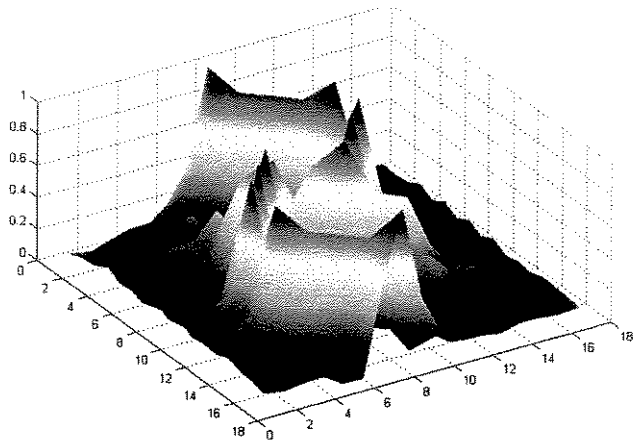


Figura 6.104 - U-matrix (3D) da configuração de neurônios apresentada na figura 6.101.

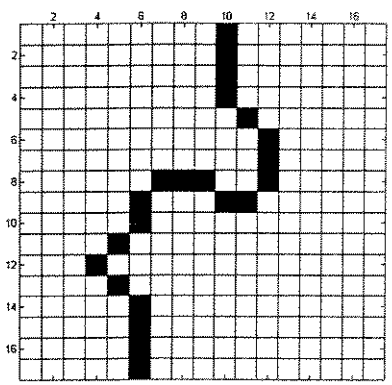


Figura 6.105 - Partição encontrada pelo algoritmo watershed (sub-mapa 2-1).

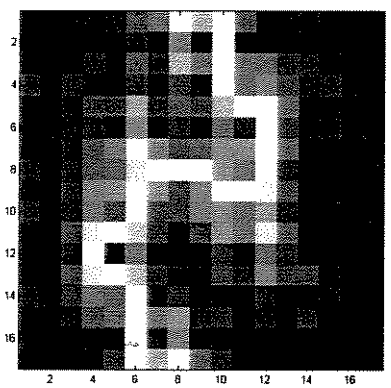


Figura 6.106 - Sobreposição das linhas de watershed sobre a imagem da U-matrix (sub-mapa 2-1).

A figura 6.107 ilustra o sub-mapa 2-1 rotulado a partir dos códigos da *U-matrix* rotulada após segmentação via *watershed*, e com a informação das classes reais mapeadas nos neurônios. A informação do mapeamento é a mesma que foi apresentada na figura 6.96. As classes $C^{2\text{-}1\text{-}1}$ e $C^{2\text{-}1\text{-}2}$ encontradas correspondem às classes reais dos dados {2} e {1} (veja a figura 6.38).

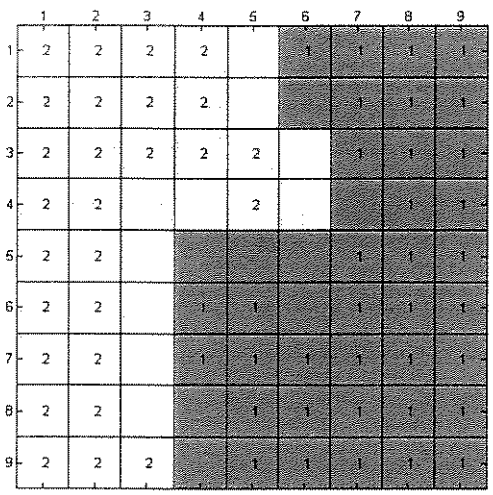


Figura 6.107 - Segmentação do sub-mapa 2-1 com as classes reais dos dados mapeadas nos neurônios

6.4.1.5 Sub-mapas 2-1-1 e 2-1-2

Os sub-mapas 2-1-1 e 2-1-2, derivados das regiões 1 e 2 do sub-mapa 2-1, foram gerados e posteriormente eliminados pelo algoritmo TS-SL-SOM. Pelo fato das estruturas de dados serem similares descreveremos apenas resultados relativos ao sub-mapa 2-1-2.

O sub-mapa 2-1-2 é filho da região 2 (classe $C^{2\text{-}1\text{-}2}$) do sub-mapa 2-1. O tamanho do sub-mapa foi 8×8. A vizinhança inicial teve raio 6 e o número máximo de iterações, no algoritmo *batch*, foi 500. O tempo de treinamento foi 1 minuto e 10 segundos (em um PC com Pentium 166 MHz) e o erro de quantização obtido foi 0.132984.

A configuração de neurônios é apresentada na figura 6.108. O histograma de vencedores é mostrado na figura 6.109 e a figura 6.110 ilustra o mapeamento efetuado pelo SOM considerando a informação das classes reais. A análise é similar à efetuada para o sub-mapa 2. As figuras 6.111 e 6.112 ilustram a *U-matrix*, respectivamente em 2 e 3-D. O gráfico de número de regiões conectadas *versus* limiar, desconsiderando o efeito dos neurônios das bordas, é apresentado na figura 6.113. Note, na figura 6.113 que apenas 35 dos 256 níveis de cinza diferiram da informação '1', resultando em $\gamma = 13.7\%$, o que está abaixo do limiar mínimo ($\gamma_{min} = 20\%$) estabelecido na seção 6.2.4. Obviamente, se estabelecêssemos um limiar menor, por exemplo 10%, poderíamos segmentar este mapa e gerar sub-mapas filhos.

Eliminando o efeito dos neurônios inativos do sub-mapa 2-1-2, temos a configuração de neurônios apresentada na figura 6.114. As figuras 6.115 e 6.116 ilustram a *U-matrix* da

configuração de neurônios apresentada na figura 6.114, em 2 e 3-D, respectivamente. Note, nas figuras 6.115 e 6.116 a existência de uma região de bordas iniciando no lado esquerdo da *U-matrix* porém cessando na parte direita da imagem. Apenas um marcador foi detectado para a *U-matrix*, tanto na figura 6.112 quanto na figura 6.116, o que não proporcionou a segmentação em mais de uma região.

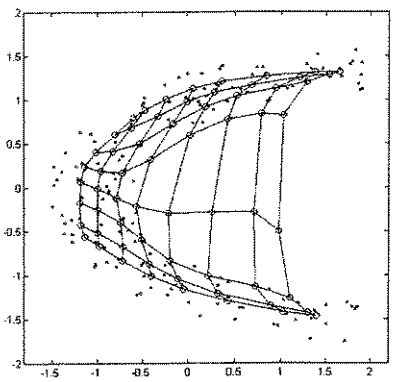


Figura 6.108 - Configuração de neurônios - sub-mapa 2-1-2.

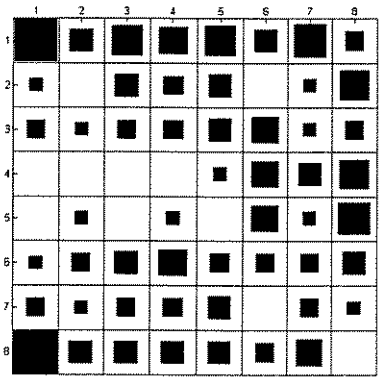


Figura 6.109 - Histograma de vencedores - sub-mapa 2-1-2.

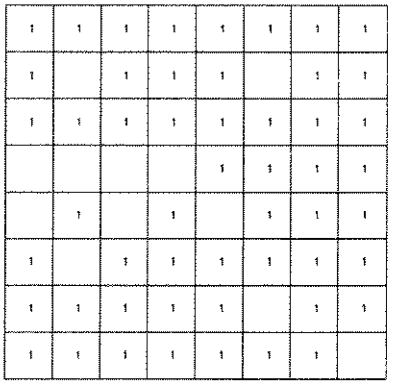


Figura 6.110 - Mapeamento das classes reais no sub-mapa 2-1-2.

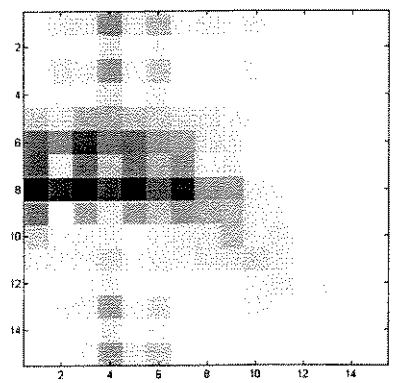


Figura 6.111 - U-matrix (2D) do sub-mapa 2-1-2.

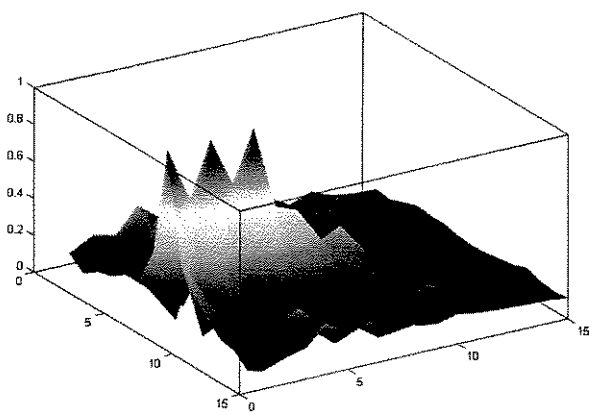


Figura 6.112 - U-matrix (3D) do sub-mapa 2-1-2.

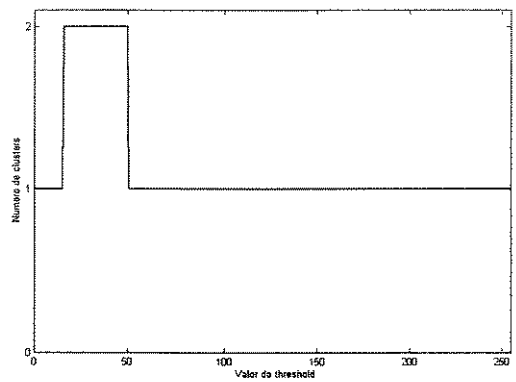


Figura 6.113 - Número de regiões conectadas versus limiar da U-matrix - sub-mapa 2-1-2 desconsiderando o efeito dos neurônios da borda do mapa.

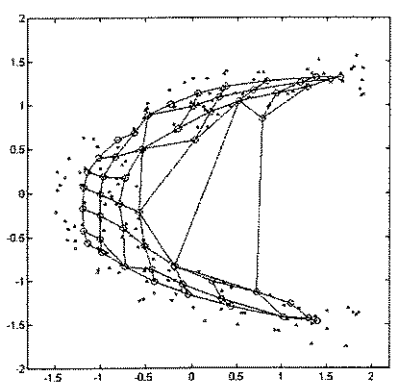


Figura 6.114 - *Dados e a configuração de neurônios (sub-mapa 2-1-2) eliminando o efeito dos neurônios inativos, $H(i, j) < 0$.*

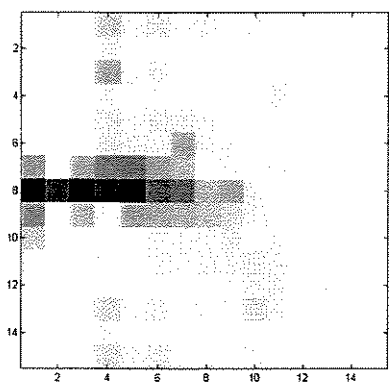


Figura 6.115 - U-matrix (2D) da configuração de neurônios apresentada na figura 6.114.

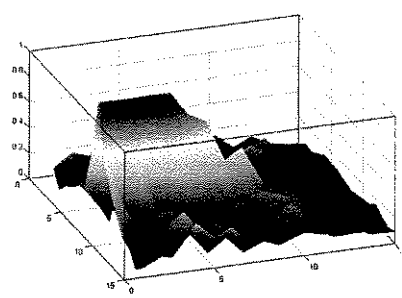


Figura 6.116 - U-matrix (3D) da configuração de neurônios apresentada na figura 6.114.

6.4.2 Conjunto de dados *Animals*

O conjunto de dados *Animals* foi descrito em Ritter & Kohonen (1989a,b) e utilizado em muitos artigos na literatura, por exemplo Adams *et al.* (1999). O objetivo é apresentar a capacidades do *TS-SL-SOM* aprender e representar a hierarquia existente nos dados.

O conjunto de dados possui 16 objetos no espaço 13-dimensional representando animais através de atributos arbitrariamente escolhidos. A tabela 6.1 ilustra o conjunto de dados, onde cada coluna é um padrão de dados. Ritter e Kohonen utilizaram um mapa bidimensional com tamanho 10 × 10. O mapa foi dividido manualmente após a auto-organização em três regiões e corresponde a pássaros, carnívoros, e herbívoros.

Resultados para este conjunto de dados usando o TS-SL-SOM foram apresentados em Costa e Netto (1999c). Foi utilizado para o mapa raiz um SOM bidimensional com topologia retangular e tamanho 7×7. A inicialização de pesos foi aleatória e o treinamento foi efetuado com o algoritmo em lote. A função de vizinhança usada foi Gaussiana e o raio inicial de vizinhança foi 5. O número de épocas foi 1000 e o tempo de treinamento sob o ambiente Matlab em um PC 166 MHz foi 50 segundos.

As figuras 6.117 e 6.118 mostram a *U-matrix* do SOM raiz, em 2 e 3-D, respectivamente. A figura 6.119 ilustra o gráfico do número de regiões conectadas em função do limiar da *U-matrix*. A solução 3 agrupamentos é vencedora, e os marcadores encontrados pelo método são apresentados na figura 6.120. Aplicando o algoritmo *watershed* usando os marcadores apresentados na figura 6.120, encontramos a partição da *U-matrix* mostrada nas figuras 6.121 e 6.122.

TABELA 6.1. CONJUNTO DE DADOS *ANIMALS* - ADAPTADO DE RITTER & KOHONEN (1989A,B).

	Dove	*Hen*	*Duck*	*Goose*	*Owl*	*Hawk*	*Eagle*	*Fox*	*Dog*	*Wolf*	*Cat*	*Tiger*	*Lion*	*Horse*	*Zebra*	*Cow*
Is																
small	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	0
medium	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
big	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
Has																
2 legs	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
4 legs	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
hair	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
hocrycs	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
mane	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
feathers	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
Likes to																
hunt	0	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0
run	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	0
fly	1	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0
swim	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

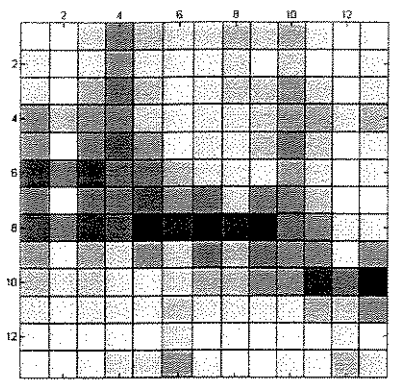


Figura 6.117 - U-matrix (2D) do mapa raiz (animals).

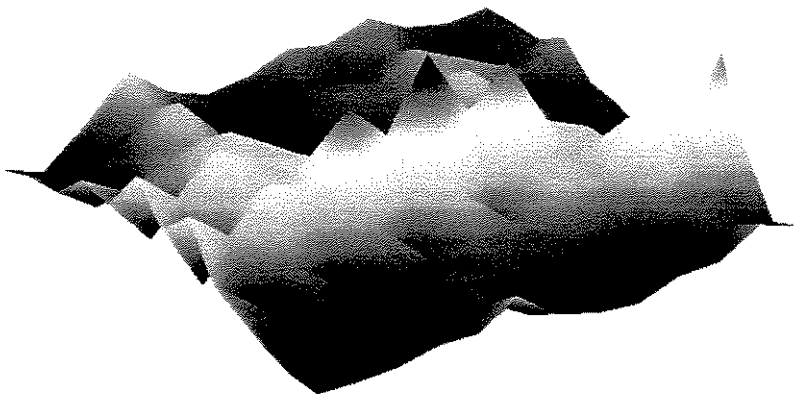


Figura 6.118 - U-matrix (3D) do mapa raiz (animals).

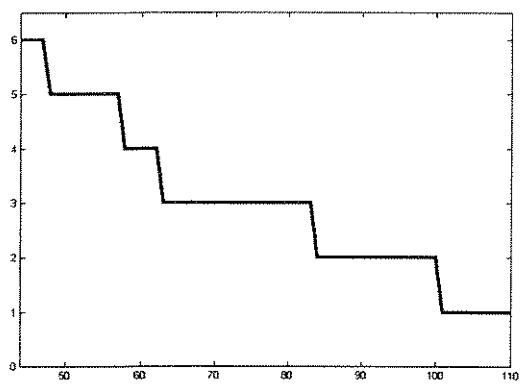


Figura 6.119 - Gráfico do número de regiões conectadas em função do limiar da U-matrix.

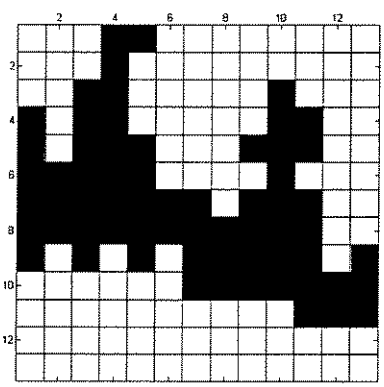


Figura 6.120 - Marcadores escolhidos pelo SL-SOM.

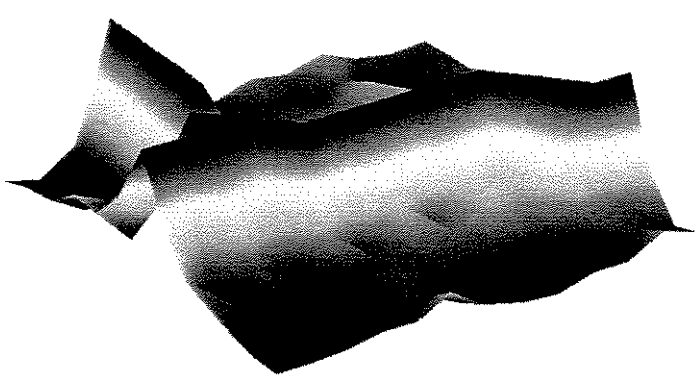


Figura 6.121 - Linhas da watershed sobrepostas à U-matrix.

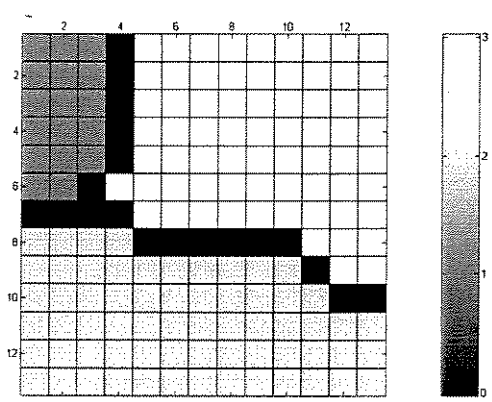


Figura 6.122 - U-matrix rotulada após segmentação via watershed.

Aplicando o procedimento do TS-SL-SOM encontramos a árvore apresentada na figura 6.123, onde cada elemento da árvore corresponde a um mapa de Kohonen. A partição de cada mapa juntamente com os nomes dos objetos (animais) sobrepostos aos neurônios vencedores é apresentada na figura 6.124.

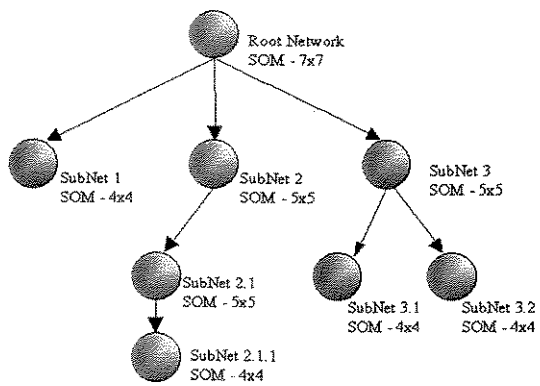


Figura 6.123 - Visão da árvore de mapas obtida

Costa e Netto (1999c) utilizaram como critério de parada da árvore uma condição bastante simples. Foi analisado o histograma cumulativo da U-matrix, H_U^C, ver figuras 6.35-6.37. Porém, neste caso o intervalo de distâncias foi dividido em apenas duas faixas, e buscou-se detectar os coeficientes angulares (a_1^1 e a_1^2) dos polinômios de primeiro grau nas faixas [*0.0*, *0.5*] e [*0.5*, *1.0*] obtidos via regressão linear. Esperava-se que a_1^1 fosse tão maior que a_1^2 quanto maior a viabilidade de haver agrupamentos. Dois valores foram utilizados, $a_1^1{}_{min} = 2$ e $a_1^2{}_{max} = 0.75$, obtidos por experimentação. Levando-se em consideração outros fatores, como a eliminação de efeitos dos neurônios das bordas, para o caso em que o mapa raiz é maior (10×10) pode-se desconsiderar o sub-mapa 2.1.1.

Os resultados apresentados na figura 6.124, para o caso do mapa raiz, são muito similares à partição manual efetuada por Ritter e Kohonen (1989a,b). O método TS-SL-SOM também obteve desempenho melhor que outras redes neurais competitivas, por exemplo a apresentada por Adams *et al.* (1999) que encontrou apenas duas sub-redes inicialmente, os mamíferos e os pássaros. Porém, como pode ser visto na figura 6.119 a solução de dois agrupamentos também foi detectada como a segunda solução mais plausível (segundo maior platô de estabilidade de regiões conectadas) pelo método. Em problemas onde a diferença percentual entre platôs $\{(N_{cr}^{k1} - N_{cr}^{k2}) / max(N_{cr}^{k1}, N_{cr}^{k2})\}$ é pequena, deveríamos escolher a solução que apresenta menor número de agrupamentos, $min(N_{cr}^{k1}, N_{cr}^{k2})$, e deixar a subdivisão para níveis posteriores da árvore.

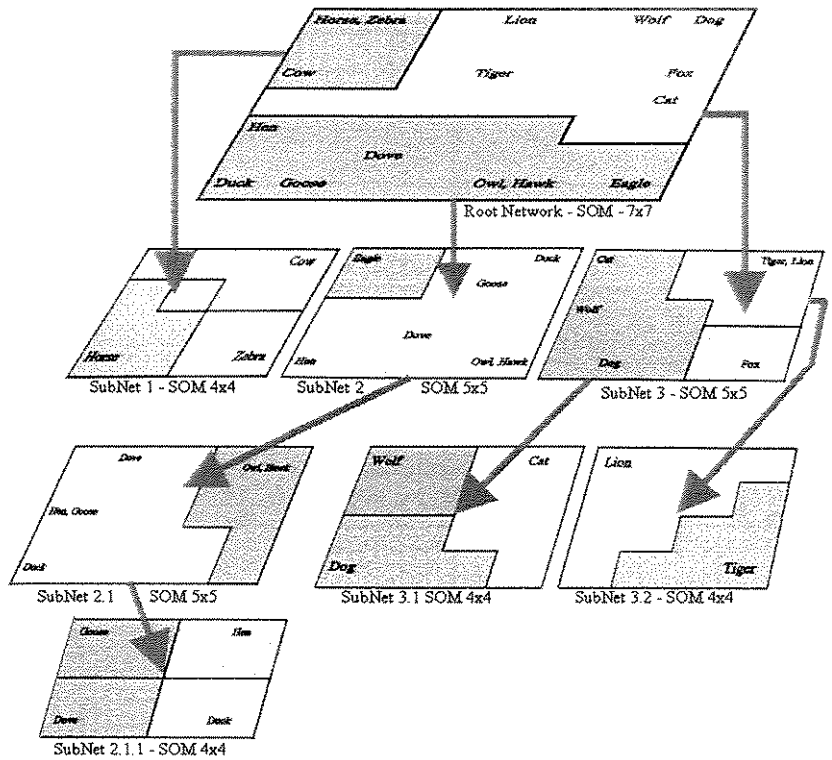


Figura 6.124 - visão da árvore de mapas obtida: as partições de SOM são apresentadas para todos os sub-mapas da árvore.

6.4.3 Conjunto de dados *Iris*

O conjunto de dados *Iris* foi originalmente usado por Anderson (1935) e Fisher (1936) e tem sido utilizado extensivamente na literatura de reconhecimento de padrões, por exemplo, em Duda e Hart (1973), Bezdek e Pal (1995), Hamad et al. (1996).

A tabela 6.2 ilustra o conjunto de dados *Iris*. Temos 50 amostras de cada uma das 3 de espécies de plantas *Iris setosa*, *Iris versicolour* e *Iris virginica*, mensuradas sob 4 variáveis que são o comprimento e largura da sépala (*sepal length* e *sepal width*) e comprimento e largura da pétala (*petal length* e *petal width*), todas medidas em centímetros. A primeira classe (*Iris setosa*) é separável linearmente das outras duas últimas (*Iris versicolor* e *Iris virginica*), que apresentam um certo grau de sobreposição. Embora saibamos o número de agrupamentos e a pertinência dos objetos às classe, desconsideraremos estas informações nos algoritmos usados, e somente a usaremos para fins de comparação com outros métodos.

O objetivo é descobrir a estrutura dos dados a partir do TS-SL-SOM. A figura 6.125 ilustra a projeção do conjunto *Iris* nos dois primeiros componentes principais. Note a sobreposição das classes 2 e 3, o que dificulta o processo de separação das classes.

Note que, apesar do método de análise de componentes principais (PCA) reduzir a dimensão, como na figura 6.125, de 4 para 2, o método é baseado em combinações lineares e tem objetivo de encontrar direções que expliquem da melhor maneira possível a variabilidade dos dados. Para este conjunto de dados, o primeiro componente corresponde a cerca de 72.8% da variabilidade total dos dados. Os outros componentes são responsáveis por 23.0%, 3.7% e 0.5%, respectivamente. Desta forma, 95.8% da variância total é explicada pelos dois primeiros componentes principais.

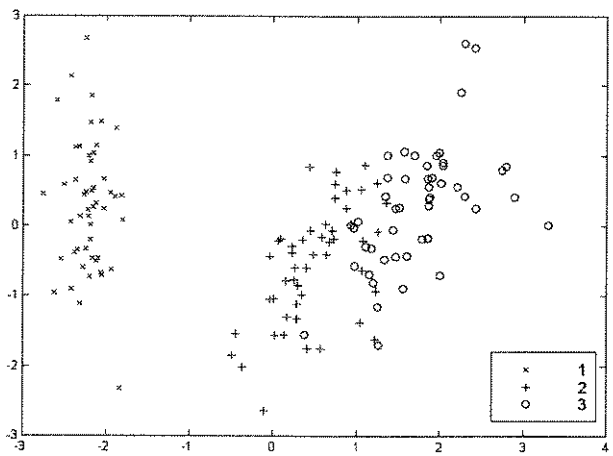


Figura 6.125 - Projeção do conjunto de dados Iris nos dois primeiros componentes principais

Foi utilizado para o mapa raiz um SOM bidimensional com topologia retangular e tamanho 10×10. A inicialização de pesos foi linear e o treinamento foi efetuado com o algoritmo em lote. A função de vizinhança usada foi Gaussiana e o raio inicial de vizinhança foi 8. O número de épocas foi 1000 e o tempo de treinamento sob o ambiente Matlab em um PC 166 MHz foi 177 segundos e o erro de quantização obtido foi 0.0171.

A figura 6.126 ilustra o mapeamento das classes 1 a 3 no mapa raiz, e a figura 6.127 ilustra o histograma de vencedores relacionado ao mapa raiz. As figuras 6.128 e 6.129 ilustram a *U-matrix* do mapa raiz. Note que apenas dois vales significativos ocorrem na U-matrix, o que é refletido no gráfico do número de regiões conectadas *versus* limiar da *U-matrix* (figura 6.130). A figura 6.131 ilustra a imagem de marcadores obtida pelo SL-SOM. As figuras 6.132 e 6.133 ilustram o histograma e o histograma cumulativo da *U-matrix*. A maior parte dos pixels está alocado para vales da *U-matrix*, e na imagem do histograma de

vencedores (figura 6.127) há uma região de neurônios inativos separando duas regiões de neurônios ativos.

A partição pelo algoritmo *watershed* é apresentada na figura 6.134. A figura 6.135 ilustra a sobreposição das linhas de *watershed* sobre a *U-matrix* em 3D.

TABELA 6.2: CONJUNTO DE DADOS IRIS. ADAPTADO DE FISHER (1936)

Iris setosa				*Iris versicolor*				*Iris virginica*			
Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width
5.10	3.50	1.40	0.20	7.00	3.20	4.70	1.40	6.30	3.30	6.00	2.50
4.90	3.00	1.40	0.20	6.40	3.20	4.50	1.50	5.80	2.70	5.10	1.90
4.70	3.20	1.30	0.20	6.90	3.10	4.90	1.50	7.10	3.00	5.90	2.10
4.60	3.10	1.50	0.20	5.50	2.30	4.00	1.30	6.30	2.90	5.60	1.80
5.00	3.60	1.40	0.20	6.50	2.80	4.60	1.50	6.50	3.00	5.80	2.20
5.40	3.90	1.70	0.40	5.70	2.80	4.50	1.30	7.60	3.00	6.60	2.10
4.60	3.40	1.40	0.30	6.30	3.30	4.70	1.60	4.90	2.50	4.50	1.70
5.00	3.40	1.50	0.20	4.90	2.40	3.30	1.00	7.30	2.90	6.30	1.80
4.40	2.90	1.40	0.20	6.60	2.90	4.60	1.30	6.70	2.50	5.80	1.80
4.90	3.10	1.50	0.10	5.20	2.70	3.90	1.40	7.20	3.60	6.10	2.50
5.40	3.70	1.50	0.20	5.00	2.00	3.50	1.00	6.50	3.20	5.10	2.00
4.80	3.40	1.60	0.20	5.90	3.00	4.20	1.50	6.40	2.70	5.30	1.90
4.80	3.00	1.40	0.10	6.00	2.20	4.00	1.00	6.80	3.00	5.50	2.10
4.30	3.00	1.10	0.10	6.10	2.90	4.70	1.40	5.70	2.50	5.00	2.00
5.80	4.00	1.20	0.20	5.60	2.90	3.60	1.30	5.80	2.80	5.10	2.40
5.70	4.40	1.50	0.40	6.70	3.10	4.40	1.40	6.40	3.20	5.30	2.30
5.40	3.90	1.30	0.40	5.60	3.00	4.50	1.50	6.50	3.00	5.50	1.80
5.10	3.50	1.40	0.30	5.80	2.70	4.10	1.00	7.70	3.80	6.70	2.20
5.70	3.80	1.70	0.30	6.20	2.20	4.50	1.50	7.70	2.60	6.90	2.30
5.10	3.80	1.50	0.30	5.60	2.50	3.90	1.10	6.00	2.20	5.00	1.50
5.40	3.40	1.70	0.20	5.90	3.20	4.80	1.80	6.90	3.20	5.70	2.30
5.10	3.70	1.50	0.40	6.10	2.80	4.00	1.30	5.60	2.80	4.90	2.00
4.60	3.60	1.00	0.20	6.30	2.50	4.90	1.50	7.70	2.80	6.70	2.00
5.10	3.30	1.70	0.50	6.10	2.80	4.70	1.20	6.30	2.70	4.90	1.80
4.80	3.40	1.90	0.20	6.40	2.90	4.30	1.30	6.70	3.30	5.70	2.10
5.00	3.00	1.60	0.20	6.60	3.00	4.40	1.40	7.20	3.20	6.00	1.80
5.00	3.40	1.60	0.40	6.80	2.80	4.80	1.40	6.20	2.80	4.80	1.80
5.20	3.50	1.50	0.20	6.70	3.00	5.00	1.70	6.10	3.00	4.90	1.80
5.20	3.40	1.40	0.20	6.00	2.90	4.50	1.50	6.40	2.80	5.60	2.10
4.70	3.20	1.60	0.20	5.70	2.60	3.50	1.00	7.20	3.00	5.80	1.60
4.80	3.10	1.60	0.20	5.50	2.40	3.80	1.10	7.40	2.80	6.10	1.90
5.40	3.40	1.50	0.40	5.50	2.40	3.70	1.00	7.90	3.80	6.40	2.00
5.20	4.10	1.50	0.10	5.80	2.70	3.90	1.20	6.40	2.80	5.60	2.20
5.50	4.20	1.40	0.20	6.00	2.70	5.10	1.60	6.30	2.80	5.10	1.50
4.90	3.10	1.50	0.20	5.40	3.00	4.50	1.50	6.10	2.60	5.60	1.40
5.00	3.20	1.20	0.20	6.00	3.40	4.50	1.60	7.70	3.00	6.10	2.30
5.50	3.50	1.30	0.20	6.70	3.10	4.70	1.50	6.30	3.40	5.60	2.40
4.90	3.60	1.40	0.10	6.30	2.30	4.40	1.30	6.40	3.10	5.50	1.80
4.40	3.00	1.30	0.20	5.60	3.00	4.10	1.30	6.00	3.00	4.80	1.80
5.10	3.40	1.50	0.20	5.50	2.50	4.00	1.30	6.90	3.10	5.40	2.10
5.00	3.50	1.30	0.30	5.50	2.60	4.40	1.20	6.70	3.10	5.60	2.40
4.50	2.30	1.30	0.30	6.10	3.00	4.60	1.40	6.90	3.10	5.10	2.30
4.40	3.20	1.30	0.20	5.80	2.60	4.00	1.20	5.80	2.70	5.10	1.90
5.00	3.50	1.60	0.60	5.00	2.30	3.30	1.00	6.80	3.20	5.90	2.30
5.10	3.80	1.90	0.40	5.60	2.70	4.20	1.30	6.70	3.30	5.70	2.50
4.80	3.00	1.40	0.30	5.70	3.00	4.20	1.20	6.70	3.00	5.20	2.30
5.10	3.80	1.60	0.20	5.70	2.90	4.20	1.30	6.30	2.50	5.00	1.90
4.60	3.20	1.40	0.20	6.20	2.90	4.30	1.30	6.50	3.00	5.20	2.00
5.30	3.70	1.50	0.20	5.10	2.50	3.00	1.10	6.20	3.40	5.40	2.30
5.00	3.30	1.40	0.20	5.70	2.80	4.10	1.30	5.90	3.00	5.10	1.80

3		3	3	3	3	3	3	3	3
3		3		3	3			3	3
2	3	3	2	2	2		3		3
2	2	2	2	2	2	2		2	2
2	2	2	2	2	2	2			
	2	2	2	2	2				1
2	2	2					1	1	1
2	2				1	1	1		1
	2			1		1	1	1	1
2			1	1	1	1	1	1	1

Figura 6.126 - Mapeamento das classes reais no mapa raiz.

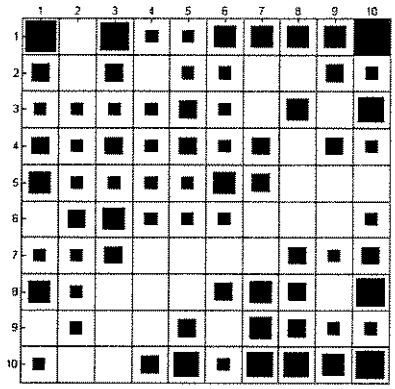


Figura 6.127 - Histograma de vencedores no mapa raiz.

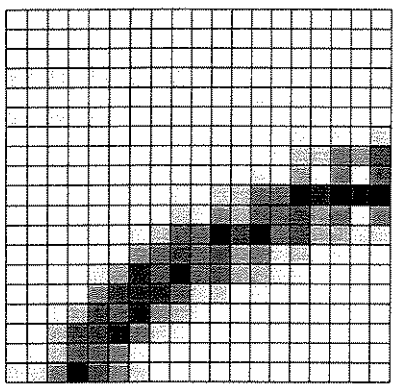


Figura 6.128 - U-matrix (2D) do mapa raiz (Iris).

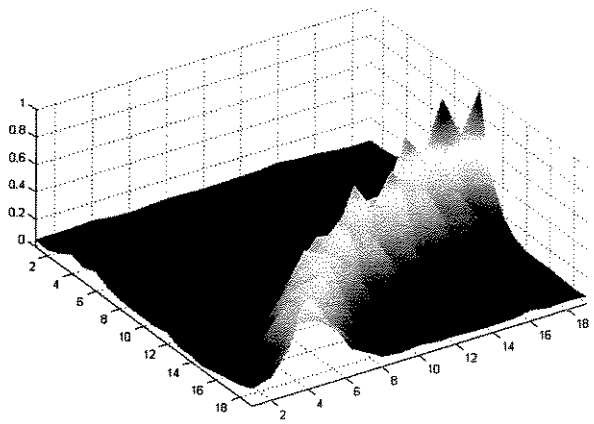


Figura 6.129 - U-matrix (3D) do mapa raiz (Iris).

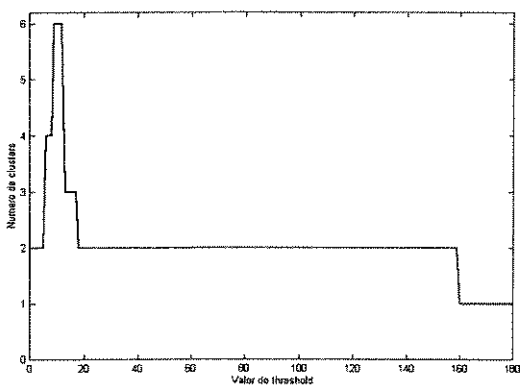


Figura 6.130 - Gráfico do número de regiões conectadas em função do limiar da U-matrix.

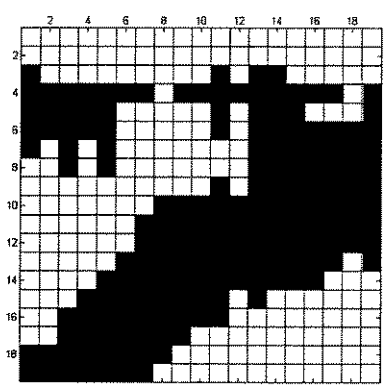


Figura 6.131 - Marcadores escolhidos pelo SL-SOM.

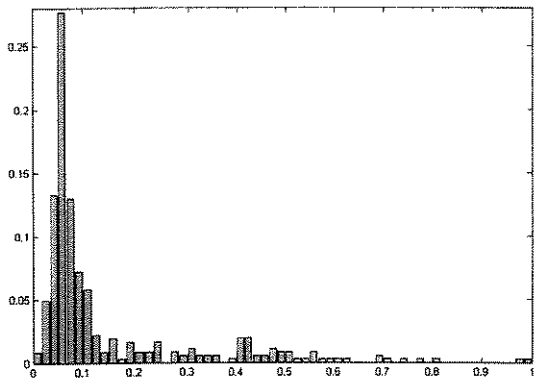


Figura 6.132 - Histograma da U-matrix.

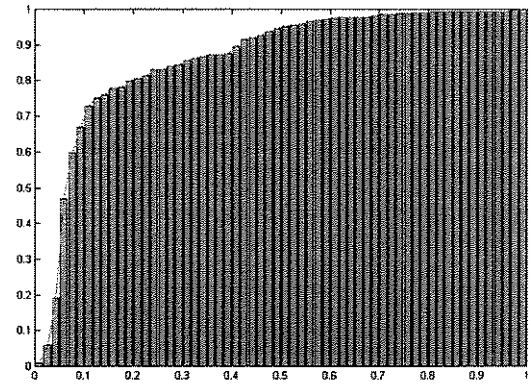


Figura 6.133 - Histograma cumulativo da U-matrix.

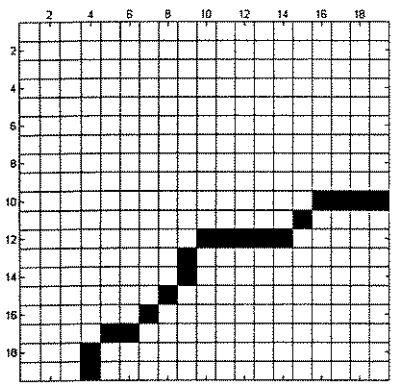


Figura 6.134 - Partição da U-matrix pela watershed.

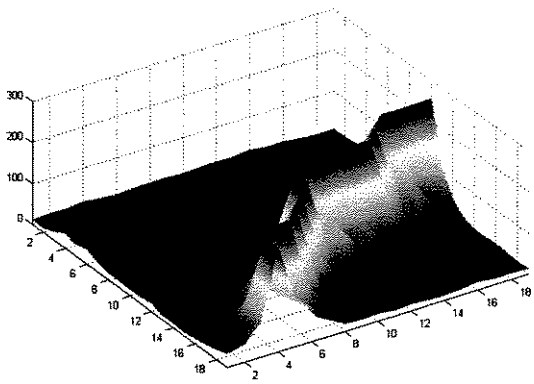


Figura 6.135 - Linhas da watershed sobrepostas à U-matrix (3D).

A figura 6.136 ilustra a partição obtida do SOM, onde temos duas regiões (agrupamentos de neurônios) correspondendo às classes de dados {2, 3} e {1}. Note, por exemplo no neurônio (2,3) o conflito, i.e., mapeamento de objetos de classes distintas sobre o mesmo neurônio.

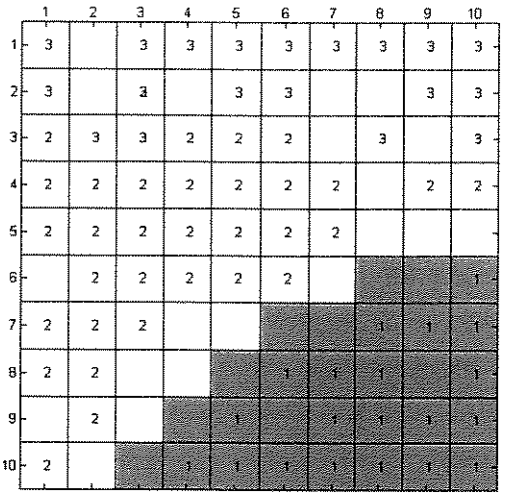


Figura 6.136 - Partição obtida do mapa raiz (conjunto de dados Iris)

6.4.3.1 Sub-mapa 1

O sub-mapa 1 é filho da região 1 (classe C^1) do mapa raiz. 100 dos 150 padrões foram classificados como pertencentes a esta região. O tamanho do sub-mapa 1 foi 9×9 e a topologia foi a mesma do mapa raiz. A vizinhança inicial teve raio 7 e o número máximo de iterações, no algoritmo *batch*, foi 1000. O tempo de treinamento foi de 108 segundos (em um PC com Pentium 166 MHz) e o erro de quantização obtido foi 0.0144.

A *U-matrix* do sub-mapa 1 é apresentada nas figuras 6.137 e 6.138. A separabilidade é mais difícil que no mapa raiz, devido à ocorrência de uma certa sobreposição nos dados. O histograma de vencedores é mostrado na figura 6.139 e a figura 6.140 ilustra o mapeamento efetuado pelo sub-mapa 1 considerando a informação das classes reais. O gráfico do número de regiões conectadas versus limiar da *U-matrix* é apresentado na figura 6.141. Note que em todos os exemplos mostrados neste capítulo (e no anterior) trabalha-se com uma versão suavizada da *U-matrix*. A figura 6.142 ilustra a ativação média do sub-mapa 1. Dois centros de ativação foram localizados nas posições (4, 4) e (6, 7).

As figuras 6.143 - 6.145 ilustram, respectivamente, a segmentação da *U-matrix* do sub-mapa 1 pelo algoritmo *watershed*, a sobreposição das linhas de *watershed* na *U-matrix* original e a *U-matrix* rotulada, que é o resultado da aplicação de codificação por regiões conectadas sobre a figura 6.143. O sub-mapa rotulado é apresentado na figura 6.146. Note que as classes reais foram mapeadas nos neurônios vencedores.

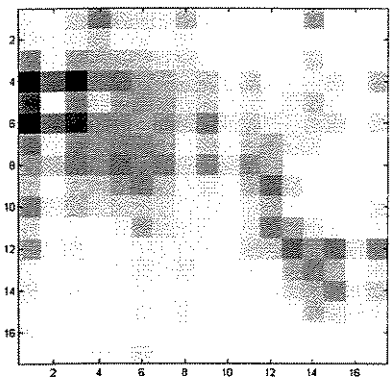


Figura 6.137 - U-matrix (2D) do sub-mapa 1 (Iris).

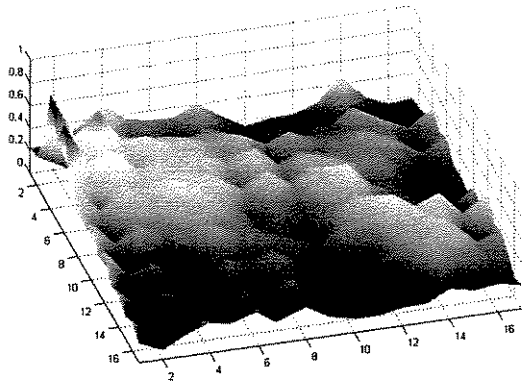


Figura 6.138 - U-matrix (3D) do sub-mapa 1 (Iris).

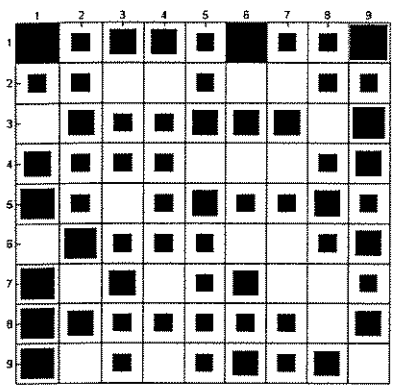
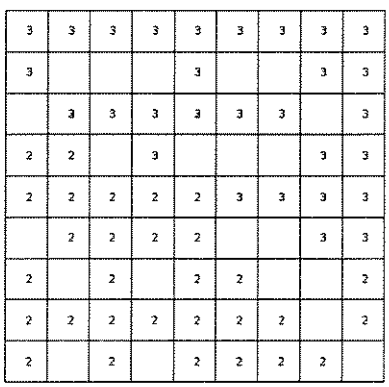


Figura 6.139 - Histograma de vencedores: sub-mapa 1 (Iris).



3	3	3	3	3	3	3	3	3
3				3			3	3
	3	3	3	3	3	3		3
2	2		3				3	3
2	2	2	2	2	3	3	3	3
	2	2	2	2			3	3
2		2		2	2			2
2	2	2	2	2	2	2		2
2		2		2	2	2	2	

Figura 6.140 - Mapeamento das classes reais no sub-mapa 1 (Iris).

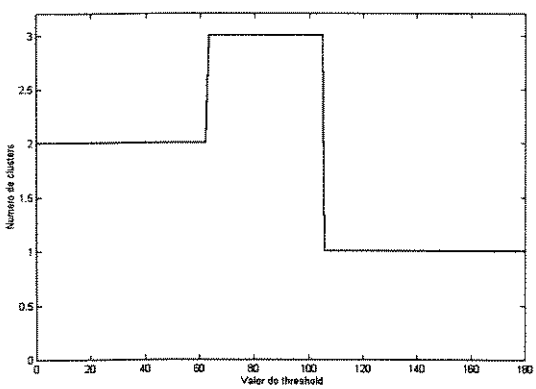


Figura 6.141 - Gráfico do número de regiões conectadas em função do limiar da U-matrix (sub-mapa 1).

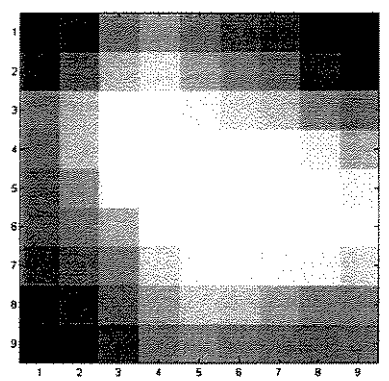


Figura 6.142 - Ativação média do sub-mapa 1(Iris)

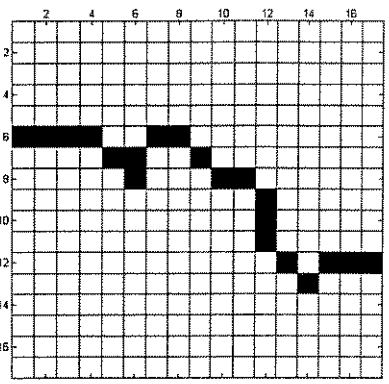


Figura 6.143 - Partição encontrada pelo algoritmo watershed (sub-mapa 1).

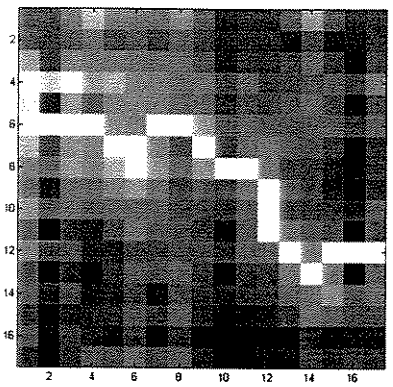


Figura 6.144 - sobreposição das linhas de watershed sobre a imagem da U-matrix (sub-mapa 1).

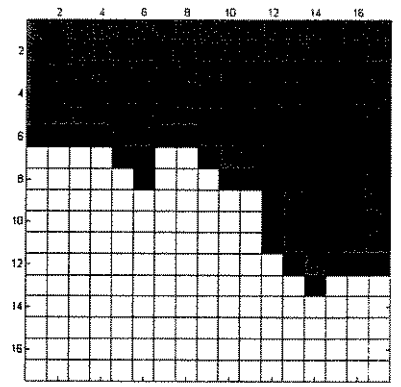


Figura 6.145 - U-matrix rotulada após segmentação via algoritmo watershed (sub-mapa 1).

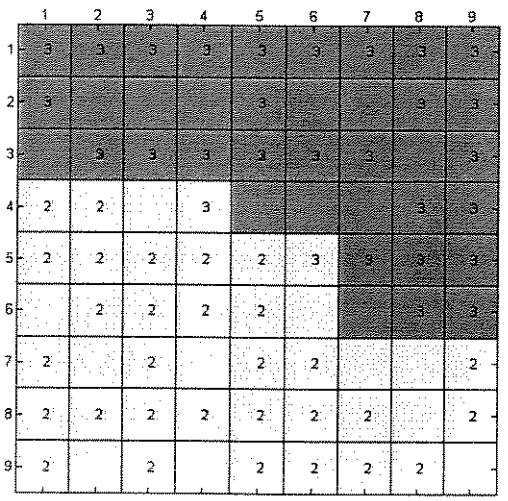


Figura 6.146 - Partição obtida do sub-mapa 1 (conjunto de dados Iris)

6.4.3.2 Sub-mapa 2

O sub-mapa 2 é filho da região 2 (classe C^2) do mapa raiz. Apenas 50 dos 150 padrões foram classificados como pertencentes a esta região. O tamanho do sub-mapa 3 foi 7×7 e a topologia foi a mesma do mapa raiz. A vizinhança inicial teve raio 6 e o número máximo de iterações, no algoritmo *batch*, foi 1000. O tempo de treinamento foi de 53 segundos (em um PC com Pentium 166 MHz) e o erro de quantização obtido foi 0.0143.

A *U-matrix* do sub-mapa 2 é apresentada nas figuras 6.147 e 6.148. O histograma de vencedores é mostrado na figura 6.149 e a figura 6.150 ilustra o mapeamento efetuado pelo sub-mapa 2, considerando a informação das classes reais. O gráfico do número de regiões conectadas *versus* o limiar da *U-matrix* é apresentado na figura 6.151. A figura 6.152

ilustra a ativação média do sub-mapa 2. Apenas um centro de ativação foi localizado na posição (4, 4) do mapa.

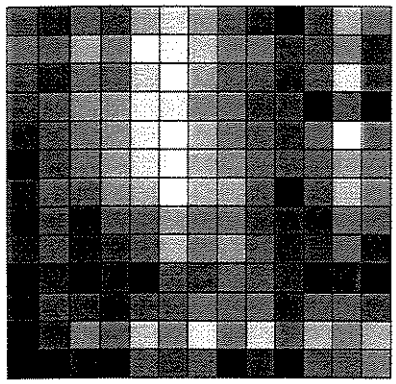


Figura 6.147 - U-matrix (2D) do sub-mapa 2 (Iris).

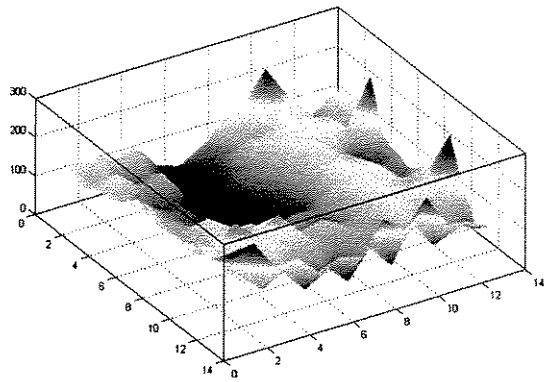


Figura 6.148 - U-matrix (3D) do sub-mapa 2 (Iris).

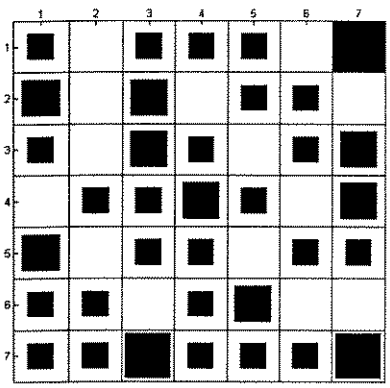


Figura 6.149 - Histograma de vencedores: sub-mapa 2 (Iris).

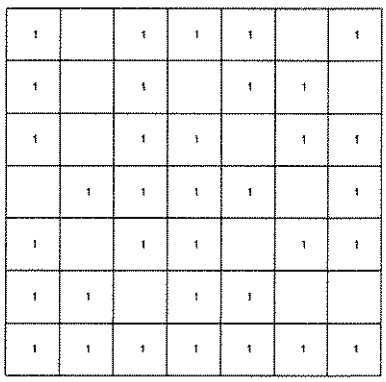


Figura 6.150 - Mapeamento das classes reais no sub-mapa 2 (Iris).

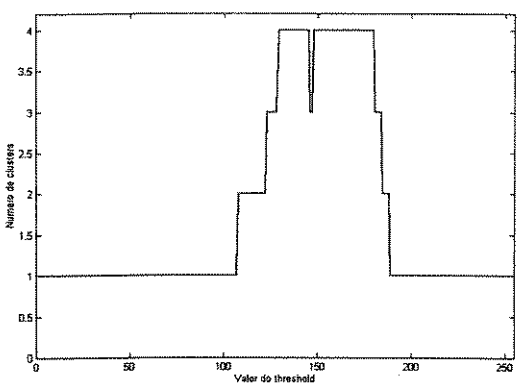


Figura 6.151 - Gráfico do número de regiões conectadas em função do limiar da U-matrix (sub-mapa 2).

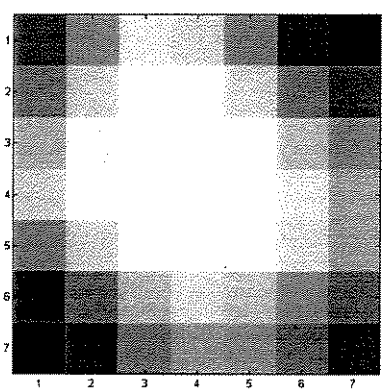


Figura 6.152 - Ativação média do sub-mapa 2(Iris)

Este mapa não foi segmentado em sub-mapas pelas seguintes razões. Primeiro, apenas um centro de ativação foi detectado (no neurônio (4, 4)). O histograma de vencedores não apresentou região significativa e conectada de neurônios inativos separando duas ou mais regiões de elevada ativação. Além disto, há outras condições como o deslocamento do gráfico do número de regiões conectadas (figura 6.151) para a direita, só indo diferir do valor 1 já próximo do nível de cinza 110. O histograma da *U-matrix*, apresentado na figura 6.153, mostra que diferentemente do ideal a maior parte das distâncias está centrada no meio do histograma. O histograma cumulativo é apresentado nas figuras 6.154 e 6.155. Efetuando regressão linear nos três intervalos, como descrito na seção 6.2.4, vemos que o coeficiente angular do segunda equação (a_1^2) é 2.16 enquanto que o da primeira equação (a_1^1) é 0.83, contrariando o estabelecido para *U-matrizes* com vales significativos, que devem apresentar $(a_1^1) > (a_1^2) > (a_1^3)$. Normalmente, quando o histograma da *U-matrix* está deslocado para a direita o histograma cumulativo fica semelhante a uma função com formato de *S*. Note no histograma de apenas 4 faixas ([0, 0.25], [0.25, 0.50], [0.50, 0.75] e [0.75, 1.0]) da figura 6.156 que temos concentração de distâncias no centro do histograma, o que diferencia bastante do caso onde há realmente agrupamentos, como no caso apresentado na figura 6.34.

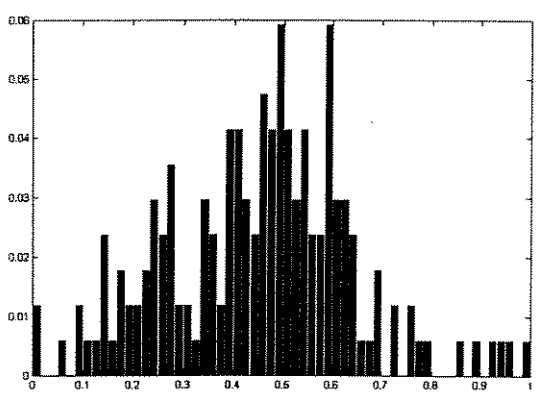


Figura 6.153 - Histograma da U-matrix (sub-mapa 2)

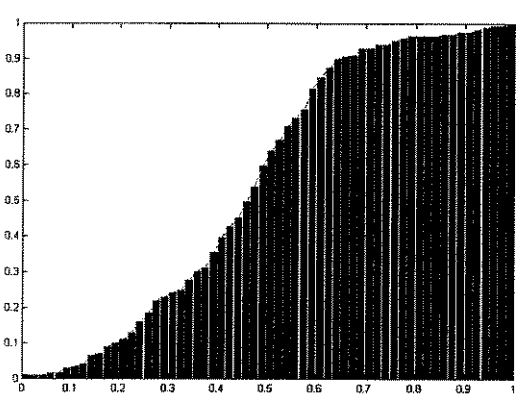


Figura 6.154 - Histograma cumulativo da U-matrix (sub-mapa 2)

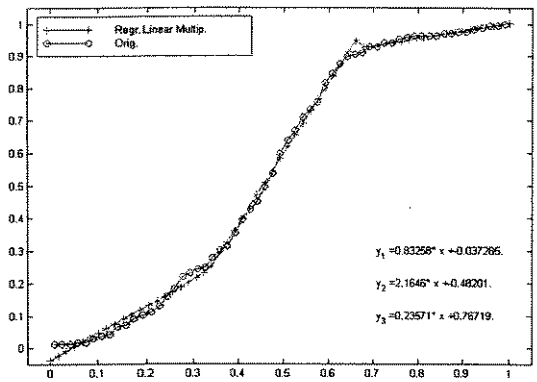


Figura 6.155 - Regressão nos três intervalos do histograma cumulativo da U-matrix (sub-mapa 2)

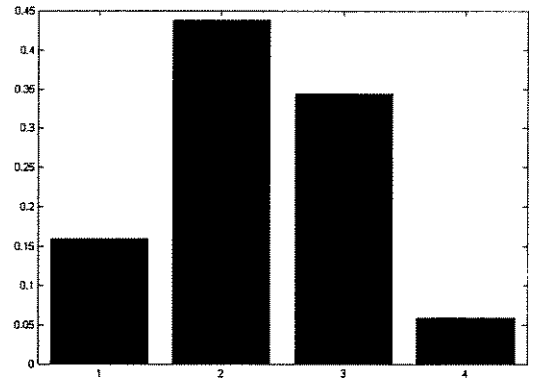


Figura 6.156 - Histograma, apenas 4 faixas, da U-matrix (sub-mapa 2)

6.4.3.3 Árvore de mapas no conjunto de dados Iris

De forma semelhante ao sub-mapa 2, os sub-mapas 1.1 e 1.2 foram criados mas eliminados devido à não detecção de sub-agrupamentos nos dados, nas condições estabelecidas. Assim, o algoritmo TS-SL-SOM detectou a estrutura apresentada na figura 6.157. No contexto de análise de agrupamentos e descoberta de conhecimento este resultado é muito bom, pois o algoritmo conseguiu detectar 2 agrupamentos separáveis no mapa raiz, separando as classes {2, 3} da classe {1} dos dados, e posteriormente separando as classes {2} e {3} no sub-mapa 1. Houve 4 erros de alocação de padrões no sub-mapa 1, o que é ótimo, pois atinge o percentual de erro dos melhores métodos atuais, por exemplo em Hamad *et al.* (1996) e Mao & Jain (1996). Este conjunto de dados em geral resulta em 15 ou 16 erros para o método *K-means*, supondo ter escolhido K igual a 3.

Pode-se interpretar o resultado da seguinte forma: existem duas classes separáveis em um primeiro passo pelo SOM, das quais uma contém 100 objetos (C^1) e a outra 50 (C^2). Focalizando atenção nestas classes, foram detectadas duas sub-classes na classe C^1 enquanto que a classe C^2 não apresentou partições. Os erros de classificação são devidos a problemas com os dados, i.e., há sobreposição dos agrupamentos, devido à má escolha dos atributos para representar as classes ou erros de medição.

Em outras simulações, usando escalonamento linear nas variáveis em vez de normalização dos dados, obtivemos 6 erros de alocação no sub-mapa 1, o que também representa um resultado excelente.

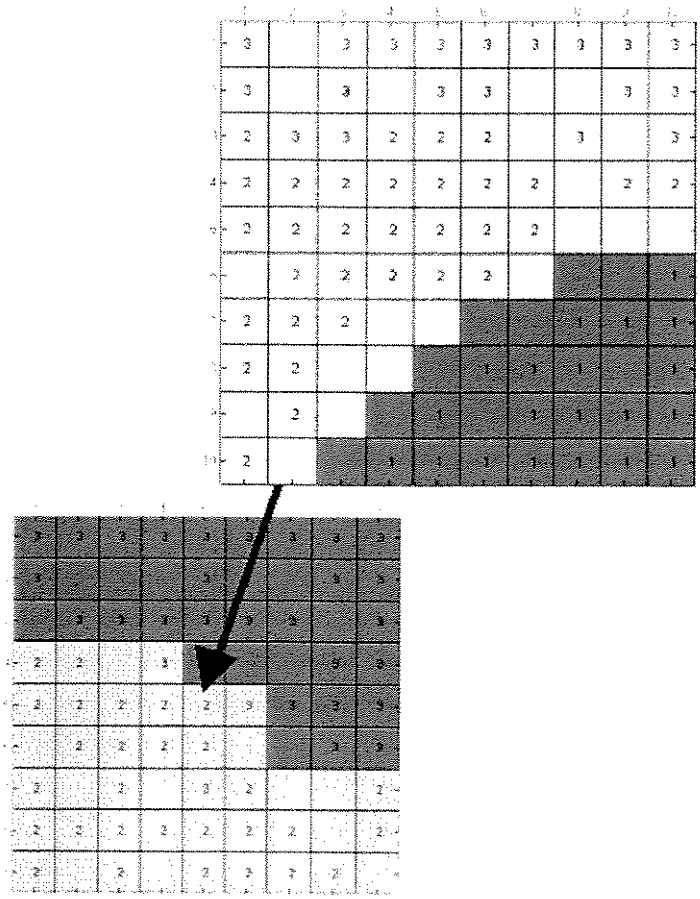


Figura 6.157 - Árvore de mapas obtida pelo TS-SL-SOM no conjunto de dados Iris.

6.5 Sumário

Descrevemos a extensão do algoritmo SL-SOM, o TS-SL-SOM que forma hierarquias de mapas de Kohonen na forma de uma árvore onde cada nó representa uma rede neural. O processo de descoberta de conhecimento do TS-SL-SOM pode ser visto como um particionamento recursivo do conjunto de dados, onde subclasses são encontradas a partir da focalização da atenção nas classes que foram detectadas pelas regiões de neurônios rotulados conjuntamente em cada mapa / sub-mapa. Aspectos importantes do TS-SL-SOM incluem sua geração automática, i.e., não especificarmos, *a priori*, o número de filhos para cada nó da árvore, e o número máximo de níveis (ou altura da árvore).

Na realidade, existe um grande espectro de possibilidades para critérios de finalização de um determinado ramo da árvore, porém este assunto está longe de estar concluído. Usamos, neste trabalho, regras do tipo IF-THEN-ELSE para testar a viabilidade de segmentar uma

U-matrix de um determinado sub-mapa. Vários critérios foram usados em conjunto, como por exemplo, a ocorrência de platô ou região de estabilidade de regiões conectadas em função do valor de limiar da *U-matrix*. Algumas das condições foram extraídas de exemplos em que se sabe *a priori* da não existência de sub-classes, como é o caso de conjuntos de dados gerados artificialmente com distribuições uniforme ou normal, onde apenas um grupo foi gerado. Foi definido a ativação média do mapa e os centros de ativação, que também estão relacionados com detecção de agrupamentos dos dados, porém devem ser usados em conjunto com outros critérios.

A estrutura obtida possui fácil interpretabilidade: dados são associados a neurônios representantes, e estes são estruturados em regiões pertencentes aos mapas constituintes da árvore. Uma vez tendo gerado a árvore de mapas, podemos também utilizar padrões novos, i.e., não usados durante o treinamento, para classificação. O padrão é inserido no mapa raiz e de acordo com o rótulo do neurônio vencedor ele é passado para os mapas filhos até atingir o mapa folha, i.e., que não possui mapas filhos. Obviamente, pelo fato de ser uma estrutura gerada de forma não supervisionada, a operação de classificação não deve ser vista como uma busca pela generalização ou maximização da taxa de acertos para um conjunto de dados para testes, como ocorre nos classificadores convencionais (Costa, 1996a-c). O objetivo é explicar grupos existentes nos dados, e suas relações hierárquicas, se houver.

Exemplos de aplicação em conjuntos de dados com estrutura conhecida (gerados artificialmente ou da literatura) foram mostrados. O TS-SL-SOM apresentou ótimos resultados e representa, na atualidade, uma ferramenta poderosa de mineração de dados e descoberta de conhecimento em grandes bases de dados.

Uma possível extensão do trabalho é o uso de redes neurais supervisionadas, por exemplo do tipo *multilayer perceptrons* ou *radial basis functions*, treinadas a partir de características extraídas de SOMs treinados em uma variedade de situações onde há ou não agrupamentos nos dados, usando informações das condições de parada da rede. Desta forma, o crescimento de novas redes TS-SL-SOM podem usar o conhecimento adquirido em uma outra rede neural no lugar de regras IF-THEN-ELSE. Outras possíveis extensões são descritas no capítulo 8.

Capítulo 7

Extensão da Análise de Agrupamentos para Mapas Auto-Organizáveis de Dimensão Maior que 2

Este capítulo apresenta outra extensão do capítulo 5: a análise de agrupamentos em uma rede SOM com espaços de saída com dimensão maior que 2. Discute-se brevemente aspectos da preservação da topologia e apresenta-se a extensão da *U-matrix* para mapas de dimensão qualquer, o *U-array*. Propõem-se métodos de análise automática do *U-array* com o objetivo da manutenção da topologia dos dados e das classes.

7.1. Introdução

O SOM forma um mapeamento de um espaço de entrada p-dimensional em um arranjo de neurônios, geralmente bidimensional, via aprendizado não supervisionado (ver capítulo 3). Como geralmente a dimensão dos dados $p >> 2$ este mapeamento realiza redução de dimensionalidade. Dois aspectos estão envolvidos no mapeamento do espaço de entrada ao espaço de saída. Primeiro, os neurônios tendem a agrupar em regiões de elevada densidade de pontos, i.e., efetuam a tarefa de quantização vetorial. Por outro lado, há uma diferença básica em relação a esquemas convencionais de agrupamentos, como o K-means: a noção de vizinhança dos neurônios e a preservação da topologia entre os espaços. Porém, atingir os dois objetivos de quantização vetorial, redução de dimensionalidade e preservação da topologia não é uma tarefa trivial.

Arranjos de neurônios em redes do tipo SOM de dimensões elevadas raramente são usados na prática por que o objetivo principal do SOM, na atualidade, é a visualização dos dados. Várias das aplicações do SOM são baseadas em mapas bidimensionais. Existem evidências de que muitas das tarefas de processamento de informações em redes neurais biológicas são efetuadas em mapas bidimensionais (Kohonen, 1997a). Porém, quando usamos o SOM com uma dimensão menor que a dimensão natural dos dados sempre haverá perda ou distorção da topologia. Por preservação da topologia podemos pensar de forma simples que as relações métricas entre os padrões no espaço original de dados devem ser mantidas no

espaço de saída, seja em uma rede do tipo SOM, ou em outro processo de redução de dimensionalidade.

A redução de dimensionalidade é estudada em estatística, principalmente sob o domínio das técnicas de escalonamento multidimensional (MDS). Um algoritmo que efetua redução de dimensionalidade preservando a topologia dos dados é uma transformação Φ: $\Re^p \rightarrow \Re^q$ que preserva a ordem de similaridade dos pontos no espaço de entrada $\Re^p$ quando estes são mapeados no espaço de saída $\Re^q$ (Flexer, 1997). Ocorre que na maioria dos algoritmos de MDS, tanto o número de vetores do espaço de entrada quanto do espaço de saída são iguais, i.e., n. Seguindo Flexer (1997, 1999), uma transformação Φ: $\hat{x} = \Phi(x)$, que preserva a similaridade impõe uma restrição do tipo $d(x_i, x_j) = \hat{d}(\hat{x}_i, \hat{x}_j)$ para todos $x_i, x_j \in \Re^p$ e todos $\hat{x}_i, \hat{x}_j \in \Re^q$, onde $i, j = 1, \ldots, n\text{-}1$, e d e $\hat{d}$ são medidas de distâncias, respectivamente, nos espaços $\Re^p$ e $\Re^q$.

Existem várias técnicas para se encontrar a transformação Φ, como por exemplo técnicas métricas de MDS (Torgerson, 1952), técnicas não métricas de MDS (Shepard, 1962a,b), a projeção de Sammon (1969) e análise de componentes principais (Jolliffe, 1986). A projeção de Sammon é obtida através da otimização, via gradiente descendente, do funcional (Flexer, 1997; Kohonen, 1997a)

$$\Phi = \frac{1}{\sum_{i=0}^{n-1}\sum_{j<i} d(x_i, x_j)} \sum_{i=0}^{n-1}\sum_{j<i} \frac{\left[d(x_i, x_j) - \hat{d}(\hat{x}_i, \hat{x}_j)\right]^2}{d(x_i, x_j)} \qquad (7.1)$$

Como exemplo, temos o caso da base de dados *Iris* (ver capítulo 6) onde o espaço original é o $\Re^4$. A figura 7.1 ilustra a projeção obtida via mapeamento de Sammon para um espaço bidimensional com um erro 0.01, obtido após 46 iterações. A figura 7.2 ilustra a projeção em duas dimensões obtida via análise de componentes principais. Mais uma vez, note que a classe 1 é separável das classes 2 e 3, que estão misturadas. Note na figura 7.1 que uma amostra da classe 2 foi projetada relativamente longe do centro da classe, o que nos faria pensar em uma amostra fora do agrupamento, caso não soubéssemos antecipadamente a pertinência do objeto à classe.

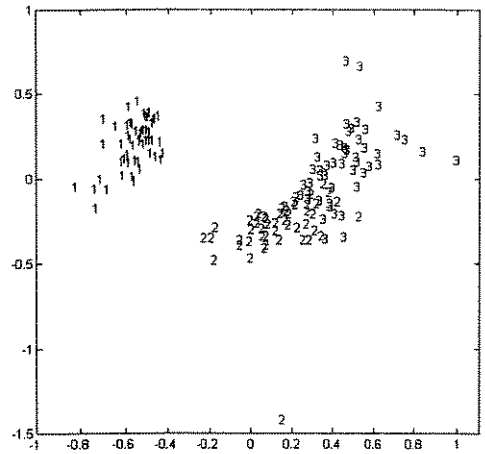


Figura 7.1 - Projeção no $\Re^2$ da base de dados Iris usando o mapeamento de Sammon.

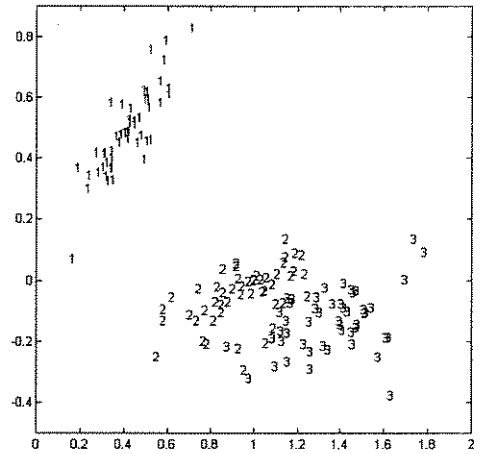


Figura 7.2 - Projeção no $\Re^2$ da base de dados Iris usando análise de componentes principais.

Note que há diferenças fundamentais entre o mapeamento de Sammon e análise de componentes principais. Enquanto o primeiro minimiza um funcional via gradiente descendente, PCA efetua uma combinação linear entre as variáveis, que geometricamente funciona como uma rotação dos eixos das variáveis, com o objetivo de capturar ao máximo a variabilidade dos dados quando usamos os componentes principais, no nosso caso o primeiro e o segundo.

Um outro exemplo é a projeção do espaço $\Re^3$ no $\Re^2$ do conjunto de dados mostrado na figura 5.58. A projeção via mapeamento de Sammon é apresentada na figura 7.3 (erro obtido 0.01), enquanto que a figura 7.4 ilustra a projeção via análise de componentes principais. Note que os resultados apresentados nas figuras 7.3 e 7.4 apresentam alguns problemas. Por exemplo, na figura 7.3 há um objeto da classe 4 que foi mapeada na classe 7 (na parte esquerda e central da figura). Ainda na figura 7.3, vemos que algumas relações de similaridades não foram preservadas. Por exemplo, a classe 8 foi mapeada no espaço de saída mais próxima da classe 1 do que o foi a classe 2, apesar da classe 2 ser mais similar à classe 1 do que a classe 8. Em relação à figura 7.4, vemos que houve uma certa sobreposição entre as classes 3 e 6, apesar de haver separação suficiente no espaço de dados original. Poderíamos supor, olhando a figura 7.4, que houvesse 7 agrupamentos de dados, caso não soubéssemos a informação das classes *a priori*, problema que agravaria mais se a dispersão dos dados nas classes fosse maior, por exemplo com os conjuntos de dados apresentados nas figuras 5.60 e 5.62.

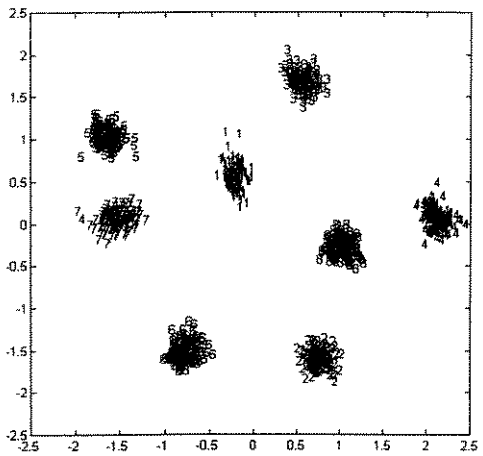


Figura 7.3 - Projeção no $\Re^2$ da base de dados apresentada na figura 5.58 usando o mapeamento de Sammon.

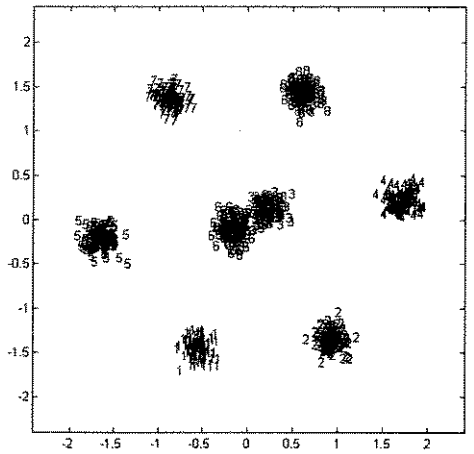


Figura 7.4 - Projeção no $\Re^2$ da base de dados apresentada na figura 5.58 usando Análise de componentes principais.

Pelo fato do SOM ter sido descrito de forma algorítmica, e não extremizar explicitamente um funcional conhecido durante o aprendizado, não há, para o SOM, uma função similar à equação (7.1), i.e., não há conexão formal entre o SOM a outro algoritmo de MDS quanto a redução de dimensionalidade. Outra diferença fundamental entre o SOM e outros algoritmos de MDS é que no SOM o número de neurônios é diferente do tamanho do conjunto de dados, n, o que pode ser pensado como uma discretização do espaço de saída.

Recentemente alguns autores têm dedicado atenção a aspectos quantitativos da preservação da topologia do SOM, como por exemplo em Bezdek e Pal (1995), Kaski e Lagus (1996), Kiviluoto (1996) e Villmann et al. (1997) e Bauer et al. (1999). A discussão efetuada neste capítulo não pretende ser exaustiva, sendo motivada basicamente pela definição do *U-array* e do método de segmentação e análise, com objetivo de agrupamento de dados e descoberta automática de conhecimento.

De forma simples, desejaríamos que dados que estejam próximos no espaço de entrada sejam mapeados em posições próximas no espaço de saída (no caso do SOM, em neurônios vizinhos ou no mesmo neurônio). Por exemplo, considere o exemplo apresentado na figura 5.58. Neste caso estamos mapeando um espaço de dimensão 3 em um mapa de dimensão 2, e a *U-matrix* obtida do *grid* de neurônios é apresentada na figura 5.59. Note que a grade bidimensional de neurônios foi contorcida pelo treinamento para atingir os dois objetivos, quantização e preservação da topologia. Apesar de termos atingido bons resultados em

detectar corretamente o número de agrupamentos e as pertinências dos objetos às respectivas classes, analisando o mapeamento efetuado notamos que classes que estão vizinhas no espaço de atributos foram mapeadas em porções distantes no mapa. Caso estivéssemos em um espaço de maior dimensionalidade sem condições de checar estas características, poderíamos cometer erros de interpretação de semelhança entre classes caso levássemos em conta as posições relativas no mapa.

A figura 7.5 ilustra o grid do mapa 15×15 após 1000 iterações do tipo batch para o problema apresentado na figura 5.58. A figura 7.6 ilustra relacionamento de vizinhança entre as classes pelo mapa após o treinamento, considerando apenas neurônios que apresentaram elevada atividade, i.e., $H(i, j) \geq 12$ vencimentos. A figura 7.7 ilustra o histograma de vencimentos dos padrões pelos neurônios, enquanto que a figura 7.8 ilustra a segmentação da *U-matrix* efetuada pelo algoritmo *watershed*. A figura 7.9 ilustra o mapa rotulado a partir dos códigos da *U-matrix* segmentada, onde as classes de dados reais (1 a 8) foram mapeadas nos neurônios vencedores.

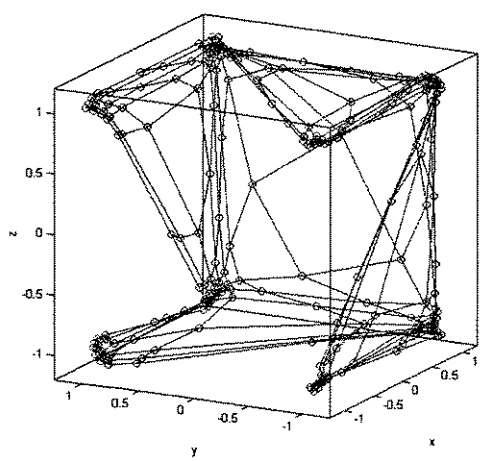


Figura 7.5: Configuração de neurônios do mapa 15×15 após 1000 iterações do algoritmo batch (problema apresentado na figura 5.58).

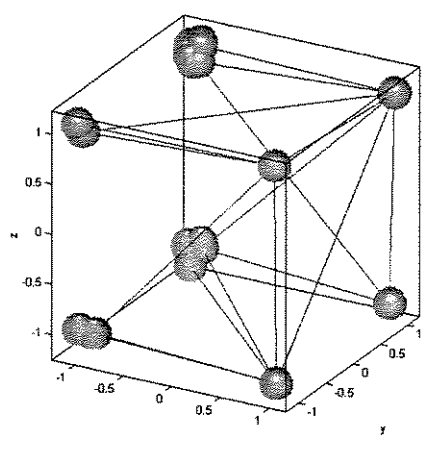


Figura 7.6: Relacionamentos entre as classes detectadas pelo SL-SOM. Apenas neurônios de elevada atividade, H(i, j) ≥ 12, foram mostrados.

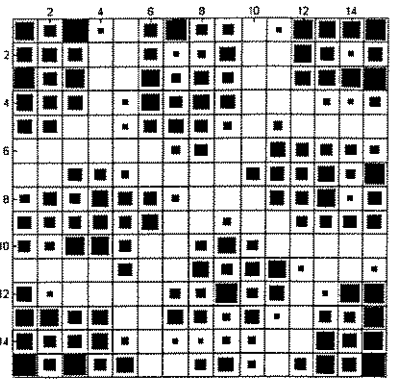


Figura 7.7 - Histograma de vencedores

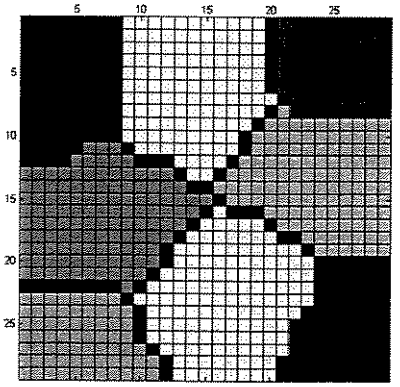


Figura 7.8 - U-matrix segmentada e rotulada após o algoritmo watershed.

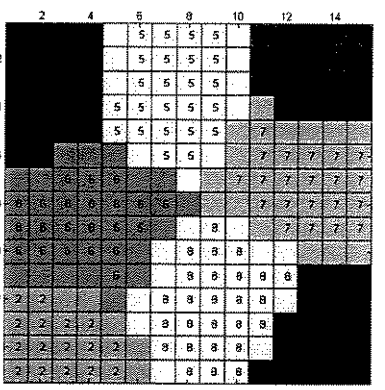


Figura 7.9 - SOM rotulado com mapeamento das classes de dados reais nos neurônios vencedores.

As classes de dados foram geradas artificialmente com estrutura conhecida, como descrito na seção 5.5.3. O conjunto de dados foi gerado com 1000 objetos contendo oito classes, na qual os vetores das médias correspondem a vértices de um cubo no espaço $\Re^3$, i.e., {(0,0,0), (0,0,1), ... (1,1,1)}, ver figura 7.10. Para o exemplo apresentado na figura 5.58 a matriz de covariâncias usada foi $\sigma^2_i \cdot I$, onde σ_i utilizado foi 0.05 e I é a matriz identidade. Desta forma, a geometria de cada uma das classes são esferas onde o parâmetro σ_i controla a variabilidade dos agrupamentos de dados.

A figura 7.11 ilustra o mapeamento das classes dos dados no SOM (note que esta figura representa informação similar à apresentada na figura 7.9). Vemos na figura 7.11 que algumas classes vizinhas no espaço de atributos (que diferem em 1 bit no vetor de médias) foram mapeadas em posições não adjacentes no mapa. Por exemplo, a classe 1 (0, 0, 0) deveria estar vizinha às classes 2 (0, 0, 1) e 3 (0, 1, 0), porém cada uma ocupa uma posição de vértice no mapa, em regiões separadas por outras regiões. Outro exemplo, a classe 2 (0, 0, 1) deveria ser vizinha da classe 4 (0, 1, 1), porém novamente estão distantes no mapa (ver figura 7.11).

Classe	Centróide
1	[0, 0, 0]
2	[0, 0, 1]
4	[0, 1, 0]
5	[0, 1, 1]
6	[1, 0, 0]
7	[1, 0, 1]
8	[1, 1, 0]
	[1, 1, 1]

Figura 7.10 - Médias (centróides) das classes geradas - vértices de um cubo.

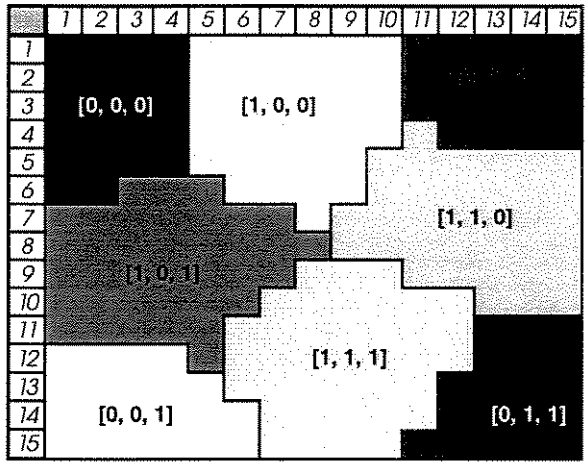


Figura 7.11 - Mapeamento das classes no SOM.

7.2 Quantificando a preservação topológica no SOM

Nesta seção consideraremos, na medida do possível, a notação empregada em Villmann et al. (1997) e em Bauer et al. (1999). Um mapa neural Ω atribui saídas $\mathbf{r}$ a entradas $\mathbf{v}$. Seja X o conjunto de entradas (padrões ou estímulos) e A o conjunto de saídas (neurônios). Consideramos, neste trabalho, o caso de quantização vetorial, onde supõe-se mapeamento *muitos-para-um.*

Em geral a qualidade do mapeamento efetuado pelo SOM é avaliada por alguns critérios, como por exemplo, o grau de continuidade do mapeamento, a resolução e a forma com que o mapeamento reflete a distribuição de probabilidade presente no espaço de entrada dos dados (Kiviluoto, 1996). Consideremos que o SOM reflete bem a distribuição de probabilidade dos dados, e concentremo-nos nos dois primeiros tópicos, que estão de uma certa forma relacionados. Kiviluoto (1996) caracterizou mapeamento contínuo quando vetores próximos no espaço de entrada são mapeados próximos no espaço de saída. Em relação à uma boa resolução, nenhum par de vetores que estejam distantes do espaço de entrada deveriam ser mapeados em posições próximas do espaço de saída.

Um aspecto importante nos parâmetros do SOM é a escolha da topologia do espaço de saída. Quando o SOM possui dimensão menor que a dimensão natural do conjunto de dados, a topologia não pode ser perfeitamente preservada, e sempre haverá um compromisso entre continuidade e resolução. O SOM tenta aproximar dimensões elevadas

contorcendo a grade elástica de neurônios na forma de uma curva Peano, resultando em descontinuidades no mapeamento. Kohonen (1997a) denominou isto de seleção automática da dimensão dos atributos. Esta propriedade é importante quando uma resolução elevada é desejada. O erro de casamento de dimensões não ocorre entre a dimensão do espaço de saída D_A e a dimensão nominal do espaço de entrada D_X, e sim entre D_A e a dimensão efetiva do espaço D_{eff} (Bauer et al., 1999).

Em algumas aplicações, a preservação da topologia do mapeamento é mais importante do que boa resolução. Seguindo Kiviluoto (1996), deveríamos usar o mapa de forma flexível a fim de encontrar, possivelmente, componentes principais não lineares do espaço de entrada, mas de forma relativamente rígida para que o mapa não se dobre, tentando representar também os componentes menos importantes. A rigidez do mapa pode ser controlada ajustando o tamanho da influência da função de vizinhança, como proposto por Speckmann *et al.* (1994).

O compromisso entre continuidade e resolução não é trivial e alguns critérios foram propostos para quantificar estas propriedades. O erro de quantização é simples de ser avaliado, sendo computado como a distância média das amostras do conjunto de dados X aos neurônios mais próximos. Para quantificar a continuidade alguns critérios foram propostos, descritos a seguir.

7.2.1 Produto Topográfico

Bauer e Pawelzik (1992) propuseram o produto topográfico P como meio de quantificar a continuidade do mapeamento, que corresponde a quantificar a preservação da topologia ou preservação da vizinhança. O método consiste em comparar os vetores de peso dos neurônios do mapa, e caso encontrem torções ou dobras no mapa, isto indica que o SOM está tentando aproximar um espaço de entrada com dimensão mais elevada, e desta forma produzindo erro topográfico.

P relaciona, para cada, neurônio a seqüência dos vizinhos no espaço de entrada à seqüência de vizinhos no espaço de saída. Seja d_X as distâncias no espaço de entrada e d_A as distâncias no espaço de saída. Para cada neurônio j, as seqüências ordenadas de distâncias d_X e d_A entre os pesos dos neurônios e índices dos neurônios, respectivamente, determinam um seqüência de vizinhos nos respectivos espaço de entrada e de saída. Seja $n_i^X(j)$ o índice do iésimo vizinho mais próximo do neurônio j no espaço de entrada, ou dos pesos, $X \subset \Re^p$ e

seja $n_i^A(j)$ o índice do i-ésimo vizinho mais próximo no espaço de saída A. O produto topográfico foi definido como

$$P = \frac{1}{N^2 - N} \sum_{j=1}^{N} \sum_{k=1}^{N-1} \log \left(\prod_{l=1}^{k} \frac{d_X(\boldsymbol{m}_j, \boldsymbol{m}_{n_i^A(j)})}{d_X(\boldsymbol{m}_j, \boldsymbol{m}_{n_i^X(j)})} \cdot \frac{d_A\left(j, n_i^A(j)\right)}{d_A\left(j, n_i^X(j)\right)} \right)^{1/2k} \tag{7.2}$$

O sinal de P indica a relação aproximada da topologia entre os espaços de entrada e saída. Valores negativos de P correspondem a um espaço de entrada com dimensão baixa, enquanto que valores de P aproximadamente zero indicam um casamento de dimensão entre os espaços. Valores de $P > 0$ indicam que o espaço de entrada tem elevada dimensionalidade em relação ao espaço de saída.

Apesar de ser um indicador interessante, P leva em consideração apenas os pesos sinápticos, e como mostrado em Villmann *et al.* (1994), o produto topográfico falha em algumas situações, provendo resultados corretos apenas quando o espaço de entrada é aproximadamente linear.

7.2.2 Coeficiente de *Spearman*

Bezdek e Pal (1995b) propuseram uma definição de preservação de topologia métrica que baseia-se na correspondência das posições de todos os pares de distâncias que ocorrem entre o espaço de entrada e o espaço de saída. A definição expressa o fato de que as posições relativas de todos os vizinhos de todo objeto devam ser preservadas. Seja Ω uma transformação métrica que preserva a topologia. Qualquer vetor sináptico $\boldsymbol{m}_r$ que tenha $\boldsymbol{m}_r'$ como k-ésimo vizinho mais próximo no espaço de entrada, $\mathbf{r}'$ deve ser o k-ésimo vizinho mais próximo de $\mathbf{r}$ no espaço de saída.

Forma-se um vetor $\boldsymbol{c}_X$ com comprimento $T = N(N\text{-}1)/2$ das distâncias entre qualquer par de neurônios no espaço de entrada, onde N é o número total de neurônios no mapa. De forma semelhante, temos o vetor $\boldsymbol{c}_A$ de distâncias no espaço de saída. Vetores de posições $\boldsymbol{b}_X$ e $\boldsymbol{b}_A$ são obtidos de $\boldsymbol{c}_X$ e $\boldsymbol{c}_A$ substituindo os componentes de $\boldsymbol{c}_X$ e $\boldsymbol{c}_A$ pelas posições respectivas na seqüência de distâncias. Assim, $\boldsymbol{b}_X(1)$ representa o índice do menor valor $\boldsymbol{c}_X$, $\boldsymbol{b}_X(2)$ o índice do segundo menor valor, etc. Bezdek e Pal propuseram que a preservação da topologia fosse quantificadas por uma medida estatística para o grau de correlação entre a ordens de posições, o coeficiente de Spearman

$$\rho(\boldsymbol{b}_X, \boldsymbol{b}_A) = 1 - \frac{1}{T^3 - T}\left[6 \sum_{k=1}^{T} (b_A(k) - b_X(k))^2 \right] \tag{7.3}$$

Uma ordenação perfeita ocorre quando $\boldsymbol{c}_X = \boldsymbol{c}_A$, e desta forma $\rho(\boldsymbol{b}_X, \boldsymbol{b}_A) = 1$. Ordenação aleatória implica em ρ aproximadamente zero, e $\rho = -1$ ocorre em mapeamentos com ordem reversa.

Bauer et al. (1999) descrevem ρ como uma medida instável e com baixa reproducibilidade, e os resultados deveriam ser tomados como uma média de várias simulações.

7.2.3 Função Topográfica

Villmann et al. (1994) propuseram a função topográfica como medida de quantificação da preservação topológica. Basicamente, o que a função topográfica faz é checar quais neurônios possuem campos receptivos adjacentes, $R_i = V_i \cap M$, onde M denota o espaço de entrada e V_i denota a célula ou poliedro do diagrama de Voronoï. Os campos receptivos R_i e R_j são considerados adjacentes caso $\overline{R}_i \cap \overline{R}_j \neq \varnothing$, onde $\overline{R}_i$ é o complemento do conjunto R_i.

A função topográfica $\Phi_L^M(s)$ é o número de neurônios que possuem campos receptivos adjacentes no espaço de entrada, porém possuem uma distância maior que s no mapa (usando a métrica *city-block*):

$$\Phi_L^M(s) = \sum_{i \in L} \#\left\{ n_j \mid j \in L, \left\| n_i - n_j \right\| > s, \;\; n_i \text{ e } n_j \text{ adjacentes} \right\} \tag{7.4}$$

onde # representa a cardinalidade de um conjunto e L é o conjunto de índices dos neurônios no mapa.

Apesar desta medida levar em consideração informações do conjunto de dados, enquanto outras como P e ρ considerarem apenas as distâncias entre os pesos sinápticos, existem alguns problemas associados à função topográfica. Por exemplo, como comparar duas funções topográficas diferentes? Outro problema é a confiabilidade da função topográfica. Da forma em que está definida ela não diferencia campos receptivos de regiões onde há grande densidade de objetos de campos receptivos onde há baixa densidade.

7.2.4 Erro Topográfico

Uma medida mais simples que a função topográfica foi proposta por Kiviluoto (1996) na forma de um simples número, ao invés de um gráfico, mesmo perdendo informações das características do mapeamento.

O erro topográfico, ξ_t é obtido considerando a adjacência dos campos receptivos e a proporção das objetos que indicam a descontinuidade local do mapeamento.

Dado um padrão $\boldsymbol{x} \in \boldsymbol{X}$, seja $\boldsymbol{m}_i$ e $\boldsymbol{m}_j$ o primeiro e o segundo vetores de pesos mais próximos de $\boldsymbol{x}$. Caso os neurônios correspondentes n_i e n_j sejam adjacentes (no espaço de saída) o mapeamento preserva, localmente, a topologia. Caso contrário, i.e., se n_i e n_j não forem adjacentes há um erro topográfico local. O erro topográfico para o mapeamento é obtido somando-se todos os erros topográficos locais para todos os padrões.

$$\xi_t = \frac{1}{n}\sum_{k=1}^{n} u(x_k), \text{ onde } u(x_k) = \begin{cases} 1, \text{ caso o primeiro e segundo neurônios vencedores sejam adjacentes} \\ 0, \text{ caso contrário.} \end{cases} \tag{7.5}$$

onde n é o número total de padrões e o fator $(1/n)$ faz com que tenhamos um valor percentual para ξ_t. Assim, ξ_t corresponde a uma proporção do número de padrões que foram mapeados corretamente. Porém, não é descrito o tipo de mapeamento incorreto. Por exemplo, não há informação caso dois padrões próximos sejam mapeados em neurônios que distem de uma unidade ou que estejam em cantos opostos do mapa.

Outras medidas, como a medida Z (Zrehen, 1993) e a medida C (Goodhill e Sejnowsky, 1997) foram discutidas em Bauer et al. (1999). Geralmente as funções que foram propostas para medir o grau de preservação da topologia possuem custo computacional elevado, principalmente se considerarmos várias simulações de vários mapas com tamanhos e dimensões diferentes, de forma a checar a melhor medida para um conjunto de dados. O número de operações para a medida P é de complexidade $O(N^3)$ e $O(N^2)$ para $\Phi_L^M(s)$ e ρ. Bauer et al. (1999) argumenta que as medidas funcionam geralmente bem, porém a medida Z é difícil de interpretar em alguns casos. A medida ρ requer uma média de vários mapas ou um controle rigoroso dos parâmetros do mapa durante o treinamento. P gerou resultados mais consistentes que outras medidas em contrapartida a um custo computacional mais elevado, e $\Phi_L^M(s)$ capta a informação tanto da configuração dos pesos quanto do conjunto de dados, porém a interpretação não é direta como índices P, ρ ou ξ_t, este último pode ser

visto como $\Phi_L^M(1)$, que também pode apresentar resultados errôneos quando os dados são ruidosos.

7.2.5 Exemplo de mapeamento de um espaço bidimensional em um SOM unidimensional

O conjunto de dados utilizado neste exemplo é semelhante ao apresentado no exemplo mostrado nas figuras 3.9 e 3.17, com a diferença de que a matriz de covariâncias, para as três classes é $\sigma^2 \cdot I$, onde σ usado foi 0.25. Note que as classes têm maior variabilidade do que no caso apresentado na figura 3.9.

O SOM usado neste exemplo é semelhante ao apresentado na figura 3.17. Temos um mapa unidimensional com 40 neurônios para representar um espaço bidimensional. A figura 7.12 ilustra o *grid* dos neurônios após 1000 iterações do algoritmo tipo lote, onde os pesos foram inicializados linearmente. A figura ilustra também o diagrama de Voronoï e os padrões, apresentados por pontos.

Note a contorção do *grid* de neurônios no espaço de dimensão maior que a dimensão do mapa para atingir o objetivo de quantização espacial. Porém, vemos que apesar das classes serem eqüidistantes no espaço de entrada, caso utilizássemos a informação do mapa, de forma automática, iríamos pensar que as classes 2 e 3 (veja seção 3.6) estão em porções distantes do espaço de atributos, pois foram mapeadas em cantos opostos do mapa.

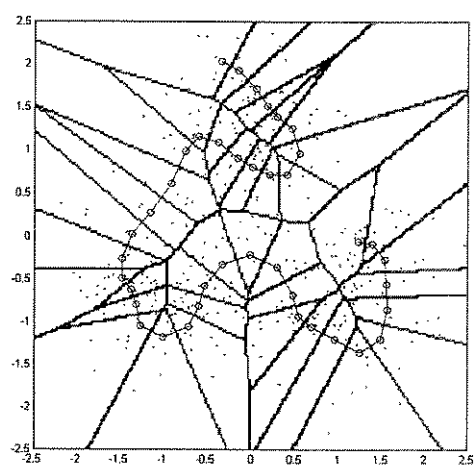


Figura 7.12: Grid de neurônios, diagrama de Voronoï e os dados, em um problema onde o espaço de entrada é bidimensional e o mapa tem topologia unidimensional.

Note na figura 7.13-a que um padrão $x' = (0.2, 0.2)$ apresentaria um erro local topográfico pois o primeiro neurônio mais próximo (primeiro vencedor ou BMU) é o neurônio 13, enquanto que o segundo vencedor (ou segundo neurônio mais próximo é o neurônio 32), veja a figura 7.13-b.

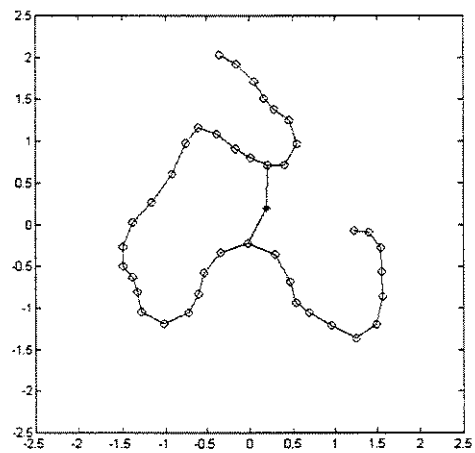


Figura 7.13-a: Um exemplo de erro topográfico local. O padrão (0.2, 0.2) possui primeiro e segundo vencedores em posições distantes no espaço de saída.

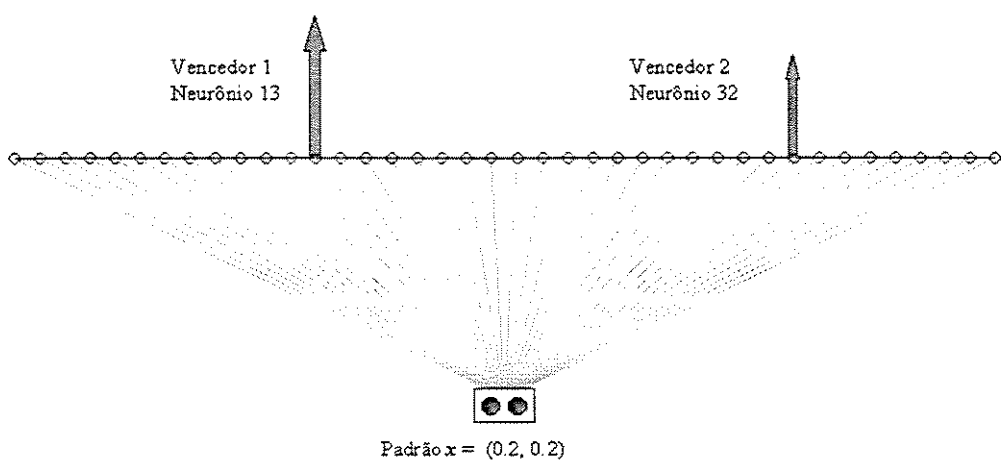


Figura 7.13-b: Um exemplo de erro topográfico local. O padrão (0.2, 0.2) possui primeiro e segundo vencedores em posições distantes no espaço de saída.

Focalizando atenção nas células do diagrama de Voronoï dos neurônios que são primeiro e segundo vencedores do padrão $\boldsymbol{x}'$, figura 7.14, vemos que tais células são adjacentes, no espaço dos atributos, porém são bastante distantes no espaço de saída.

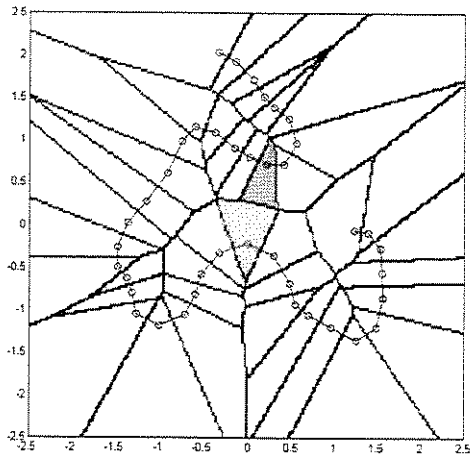


Figura 7.14 - Células de Voronoï adjacentes decorrentes de dois neurônios não vizinhos no mapa.

A figura 7.15 ilustra a configuração de neurônios e as regiões no espaço dos atributos onde há erro topográfico local, i.e., o primeiro e segundo vencedores não são adjacentes no espaço de saída do mapa. Na figura 7.16 também são mostrados os dados utilizados no treinamento, além da informação apresentada na figura 7.15. Note que apenas os objetos que estão nas regiões marcadas com preto (ver figura 7.15) que contribuem para o erro topográfico. Desta forma, caso nenhum ponto caísse nestas regiões marcadas na figura 7.15, não seria detectado nenhum erro se usássemos a equação apresentada na seção 7.2.4.

Pode-se definir um erro topográfico global (ξ_{tG}), como a integral no espaço do erro topográfico local. Idealmente ξ_{tG} deveria ser nulo. Um exemplo em que as dimensões do espaço de entrada e de saída são iguais, por exemplo em uma região quadrada bidimensional, o erro topológico global é nulo, veja a figura 7.17.

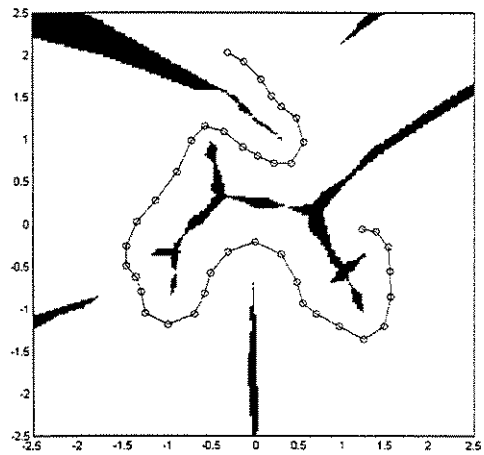


Figura 7.15 - Configuração de neurônios e as regiões no espaço que possuem erro topográfico local.

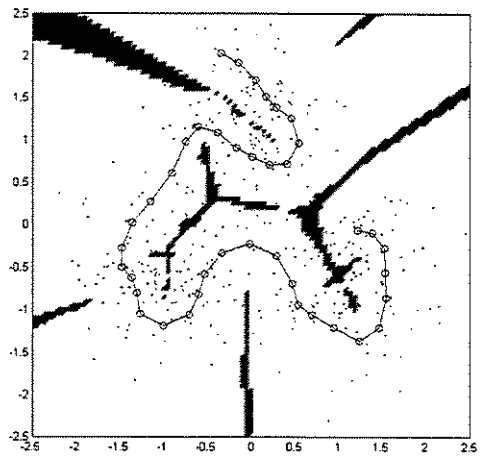


Figura 7.16 - Configuração de neurônios, regiões no espaço que possuem erro topográfico local e os dados utilizados no treinamento.

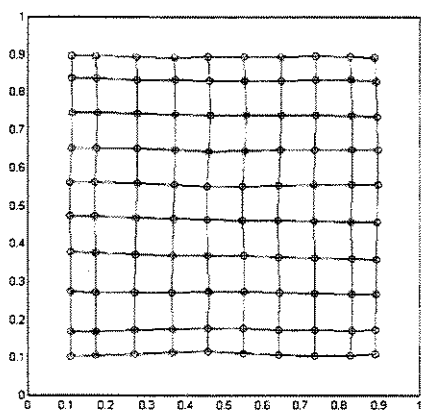


Figura 7.17 - Erro topográfico nulo para o exemplo apresentado na figura 6.3

7.2.6 Mapeamento de um espaço tridimensional em um SOM bidimensional

Vimos na seção 7.1 que mapeando classes do $\Re^3$ em um mapa bidimensional pode ocasionar um erro topográfico, ou perda na preservação da topologia, devido ao processo de curvatura da grade elástica definida em uma topologia com dimensão inferior à do espaço de entrada. Porém o quanto haverá de perda não é simples de quantificar, como vimos nas medidas propostas (seções 7.2.1 - 7.2.4).

Por exemplo, caso tenhamos um problema em que existem três agrupamentos, por exemplo hiperesféricos e com variabilidade tal que não haja sobreposição entre as classes, e que suas

médias estejam ao longo de uma reta. Caso utilizemos um SOM unidimensional a informação da topologia entre as classes será mantida, mesmo que tenhamos erros topográficos locais, devido à quantização efetuada pelo SOM.

Isto pode ser estendido para outras dimensões. Por exemplo, no caso do espaço $\Re^3$ onde as médias das classes definem um plano, poderemos utilizar um mapa bidimensional. A figura 7.18 ilustra um conjunto de dados similar ao apresentado na figura 5.58, porém apenas 4 classes foram usadas. Note que os centros das classes formam um plano imaginário que é capturado pela inicialização linear do SOM (figura 7.19). A configuração de neurônios obtida a partir da inicialização linear possui erro topográfico zero (veja a figura 7.19) porém o erro de quantização é elevado.

A figura 7.20 ilustra os dados e a configuração do mapa bidimensional, com tamanho 12×12, após 1000 iterações do algoritmo batch. Neste exemplo os quatro agrupamentos foram detectados e a informação de vizinhança das classes foi mantida, diferentemente do caso apresentado na seção 7.1 (ver figura 7.11). Porém, focalizando atenção em cada região de neurônios alocados para representar os agrupamentos notam-se curvaturas locais do mapa que produzem erros topográficos locais (figura 7.21). Para este exemplo, dos 500 padrões apenas 7 produziram erro topográfico local, o que corresponde a $\xi_t = 1.4\%$. Porém, a topologia das classes foi preservada. À medida que as classes deixam de ter suas médias sob um plano imaginário o erro topográfico aumenta e a topologia entre as classes também é gradualmente perdida, como ocorreu no problema ilustrado nas figuras 7.5 - 7.11.

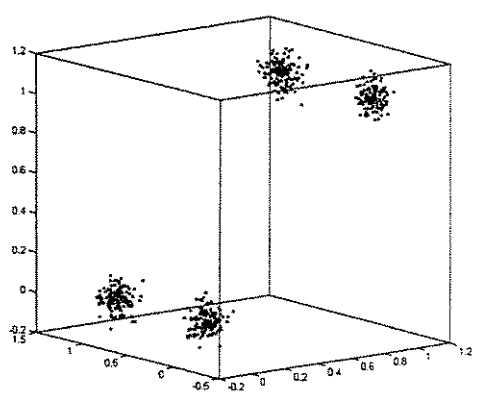


Figura 7.18 - Conjunto de dados - quatro agrupamentos no $\Re^3$

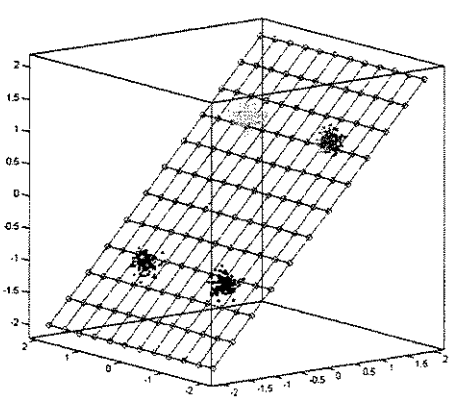


Figura 7.19 - Inicialização linear de um SOM bidimensional com tamanho 12×12 usando os dados da figura 7.14.

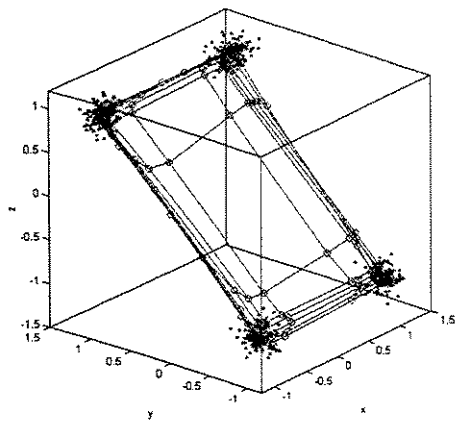


Figura 7.20 - Configuração de neurônios e dados após 1000 iterações do algoritmo batch

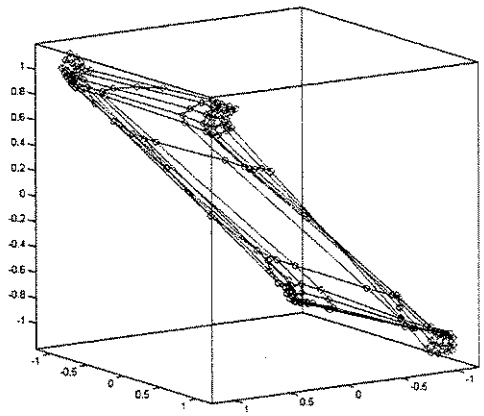


Figura 7.21 – Visualização do grid de neurônios após treinamento. Note a curvatura local sobre os agrupamentos de dados (vértices do mapa). A topologia das classes foi preservada.

7.3 Extensão da *U-matrix* a mapas de elevada dimensão: o *U-array*

O *U-array* é proposto como uma extensão da *U-matrix* (ver capítulo 5). Para o caso de SOM com dimensão do espaço de saída 3 podemos ainda visualizar as relações entre os neurônios efetuando-se cortes no *array* tridimensional. Para dimensões maiores do espaço de saída, apesar de não podermos visualizar em uma única imagem as relações entre os neurônios, podemos escolher combinações de pares de componentes e visualizar imagens bidimensionais como projeções do espaço de maior dimensão.

Nesta seção, descreveremos a extensão para o caso em que o SOM tem espaço de saída tridimensional. A extensão para casos de maior dimensão pode ser facilmente efetuada. Idealmente deveríamos escolher a dimensão do espaço de saída do SOM através de informações provenientes diretamente dos dados. Pode-se usar a informação da dimensão fractal, porém toda esta área ainda carece de maiores estudos (Bauer et al., 1999). A estratégia adotada na prática é escolher uma dimensão para o mapa e para os outros parâmetros, e ao final do treinamento pode-se checar índices como o produto topográfico, o erro topográfico ou a função topográfica. Caso os índices estejam dentro de valores aceitáveis (ver seção 7.2) pode-se dar início à análise do SOM pela *U-matrix* ou pelo *U-*

array. Caso contrário, deveríamos reiniciar o treinamento com um SOM com maior ou menor dimensão, de acordo com as informações obtidas a partir dos índices (ver seção 7.2).

No caso do *U-array* onde o SOM tem espaço de saída tridimensional, temos, além das distâncias *dx*, *dy* e *dxy*, as distâncias *dz*, *dxz*, *dyz* e *dxyz*. A figura 7.22 ilustra um SOM com tamanho 3×3×3. Note que podemos pensar em vários mapas bidimensionais contidos neste mapa. As figuras 7.23 - 7.25 ilustram tais mapas. Para facilitar a visualização das relações entre os neurônios, apenas o espaço de saída está sendo apresentado, porém, supõe-se que todos os neurônios estejam conectados à camada de entrada por pesos sinápticos.

Note que poderíamos pensar em várias *U-matrizes*, uma para cada um dos nove mapas bidimensionais apresentados nas figura 7.23 - 7.25. Considere um mapa retangular de tamanho $X \times Y \times Z$, no caso tridimensional. Da mesma forma que na *U-matrix*, o *U-array* terá tamanho $(2X\text{-}1) \times (2Y\text{-}1) \times (2Z\text{-}1)$. Veja que para o mapa 3×3×3 apresentado na figura 7.22 teremos um *U-array* de tamanho 5×5×5, e poderíamos usar a informação das *U-matrizes* de cada mapa (veja figuras 7.23-7.25) e interpolar nas posições interiores do *U-array*.

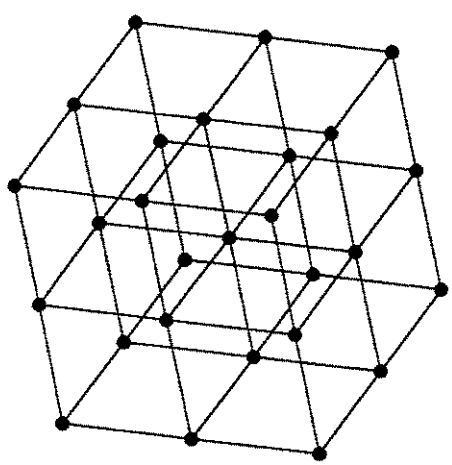


Figura 7.22 - Espaço de saída de uma rede SOM com tamanho 3×3×3.

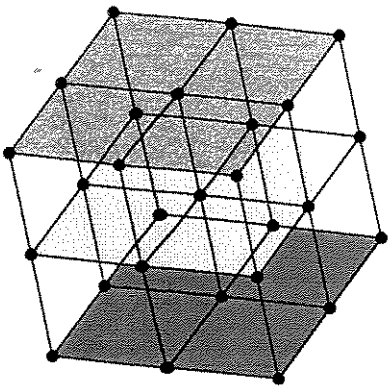


Figura 7.23 - Mapas bidimensionais considerando os planos nas direções X e Z do SOM com tamanho 3×3×3 (Y = 1, 2 ou 3)

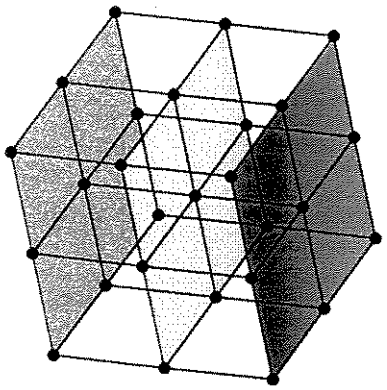


Figura 7.24 - Mapas bidimensionais considerando os planos nas direções Y e Z do SOM com tamanho 3×3×3 (X = 1, 2 ou 3)

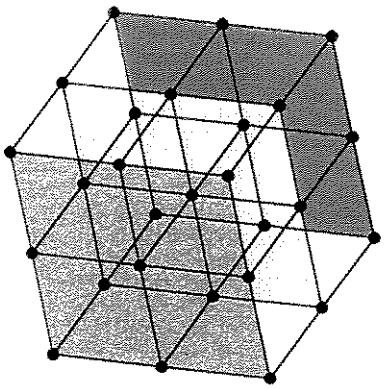


Figura 7.25 - Mapas bidimensionais considerando os planos nas direções X eY do SOM com tamanho 3×3×3 (Z = 1, 2 ou 3)

As distâncias entre os neurônios podem ser vistas de forma mais simples para um mapa com tamanho 2×2×2, como o apresentado na figura 7.26, o que resulta em um *U-array* de tamanho 3×3×3.

As distâncias *dx*, *dy* e *dz* (equações 7.6 - 7.8) também são mostradas na figura 7.26. A notação usada nesta seção é a mesma da seção 5.1. Seja $[b_{x,y,z}]$ a matriz de neurônios e $[wi_{x,y,z}]$ a matriz de pesos. As distâncias *dxy*, *dxz* e *dyz* são semelhantes à distância usada no caso bidimensional, *dxy*, apresentada pela equação 5.3. As fórmulas para tais distâncias são apresentadas nas equações 7.9 - 7.11, e ilustradas nas figuras 7.27 - 7.29, respectivamente.

$$dx(x, y, z) = \left\| b_{x,y,z} - b_{x+1,y,z} \right\| = \sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x+1,y,z}} \right)^2} \tag{7.6}$$

$$dy(x, y, z) = \left\| b_{x,y,z} - b_{x,y+1,z} \right\| = \sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x,y+1,z}} \right)^2} \tag{7.7}$$

$$dz(x, y, z) = \left\| b_{x,y,z} - b_{x,y,z+1} \right\| = \sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x,y,z+1}} \right)^2} \tag{7.8}$$

$$dxy(x,y,z) = \frac{1}{2}\left(\frac{\left\|b_{x,y,z} - b_{x+1,y+1,z}\right\|}{\sqrt{2}} + \frac{\left\|b_{x,y+1,z} - b_{x+1,y,z}\right\|}{\sqrt{2}}\right)$$

$$= \frac{1}{2\sqrt{2}}\left[\sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x+1,y+1,z}}\right)^2} + \sqrt{\sum_i \left(w_{i_{x,y+1,z}} - w_{i_{x+1,y,z}}\right)^2}\right] \quad (7.9)$$

$$dxz(x,y,z) = \frac{1}{2}\left(\frac{\left\|b_{x,y,z} - b_{x+1,y,z+1}\right\|}{\sqrt{2}} + \frac{\left\|b_{x,y,z+1} - b_{x+1,y,z}\right\|}{\sqrt{2}}\right)$$

$$= \frac{1}{2\sqrt{2}}\left[\sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x+1,y,z+1}}\right)^2} + \sqrt{\sum_i \left(w_{i_{x,y,z+1}} - w_{i_{x+1,y,z}}\right)^2}\right] \quad (7.10)$$

$$dyz(x,y,z) = \frac{1}{2}\left(\frac{\left\|b_{x,y,z} - b_{x,y+1,z+1}\right\|}{\sqrt{2}} + \frac{\left\|b_{x,y+1,z} - b_{x,y,z+1}\right\|}{\sqrt{2}}\right)$$

$$= \frac{1}{2\sqrt{2}}\left[\sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x,y+1,z+1}}\right)^2} + \sqrt{\sum_i \left(w_{i_{x,y+1,z}} - w_{i_{x,y,z+1}}\right)^2}\right] \quad (7.11)$$

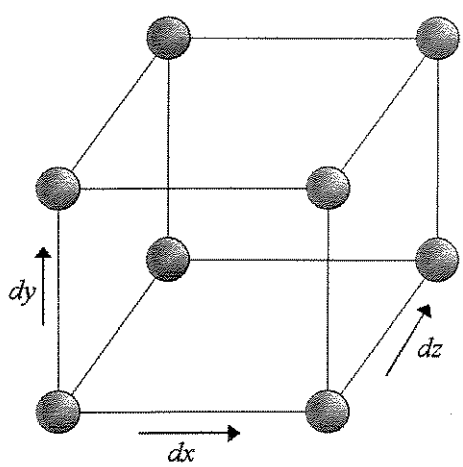


Figura 7.26 - Ilustração de um mapa com tamanho 2×2×2 e as distâncias dx, dy e dz.

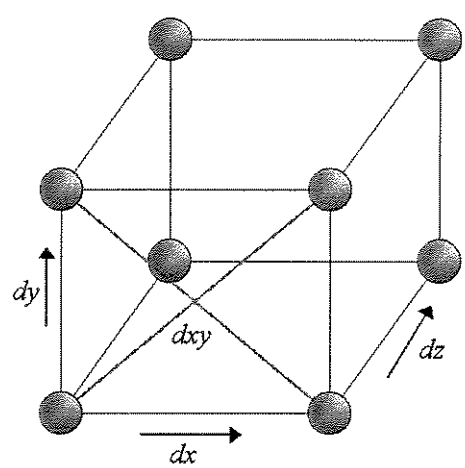


Figura 7.27 - Ilustração do mapa de tamanho 2×2×2 e as distâncias dxy, dx, dy e dz.

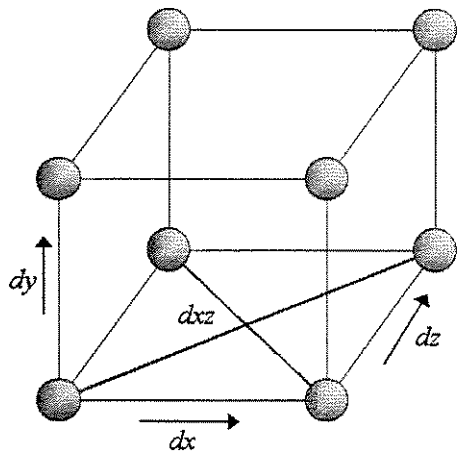


Figura 7.28 - Ilustração do mapa de Tamanho 2×2×2 e as distâncias dxz, dx, dy e dz.

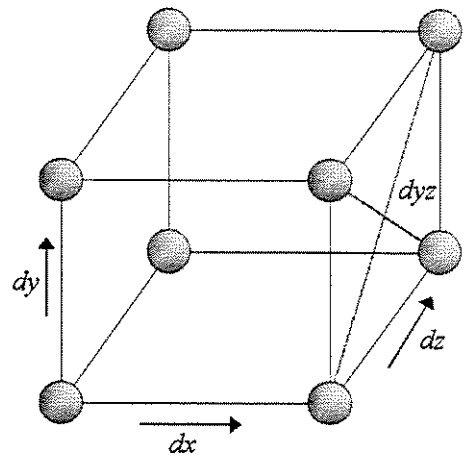


Figura 7.29 - Ilustração do mapa de tamanho 2×2×2 e as distâncias dyz, dx, dy e dz.

A distância *dxyz* é uma média das distâncias entre os vértices opostos do cubo (veja a figura 7.30). Um esquema contendo exemplos das distâncias (apenas uma de cada) é apresentada na figura 7.31, onde aparece também o endereço ou índice do neurônio no espaço de saída da rede. Note que várias distâncias não foram inseridas no exemplo, por questões de visibilidade, e que a complexidade e número de distâncias diferentes aumenta com a dimensão do mapa.

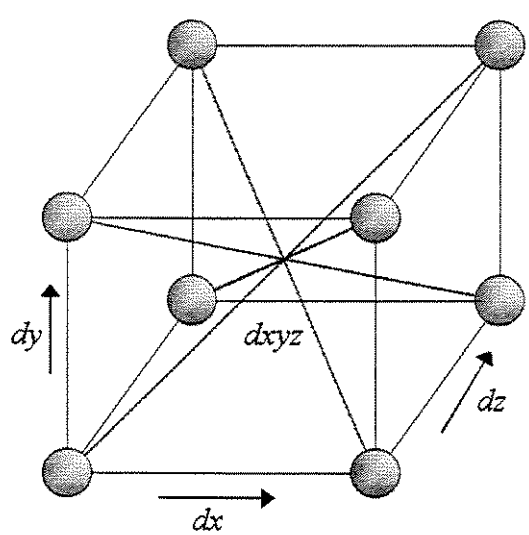


Figura 7.30 - Ilustração do mapa de tamanho 2×2×2 e as distâncias dxyz, dx, dy e dz.

$$dxyz(x,y,z) = \frac{1}{4\sqrt{3}}\left(\left\|b_{x,y,z} - b_{x+1,y+1,z+1}\right\| + \left\|b_{x,y,z+1} - b_{x+1,y+1,z}\right\| + \left\|b_{x,y+1,z+1} - b_{x+1,y,z}\right\| + \left\|b_{x,y+1,z} - b_{x+1,y,z+1}\right\|\right) \tag{7.12-a}$$

$$dxyz(x,y,z) = \frac{1}{4\sqrt{3}}\left[\sqrt{\sum_i \left(w_{i_{x,y,z}} - w_{i_{x+1,y+1,z+1}}\right)^2} + \sqrt{\sum_i \left(w_{i_{x,y,z+1}} - w_{i_{x+1,y+1,z}}\right)^2} + \right.$$
$$\left. + \sqrt{\sum_i \left(w_{i_{x,y+1,z+1}} - w_{i_{x+1,y,z}}\right)^2} + \sqrt{\sum_i \left(w_{i_{x,y+1,z}} - w_{i_{x+1,y,z+1}}\right)^2}\right] \tag{7.12-b}$$

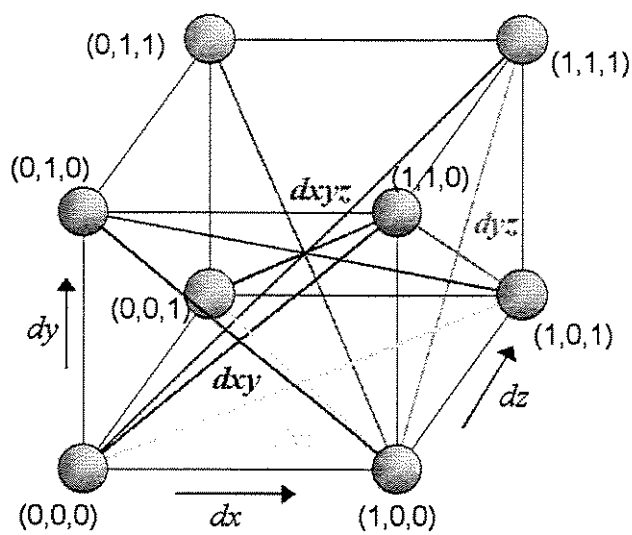


Figura 7.31 - Ilustração do mapa de tamanho 2×2×2 com exemplos de distâncias

Considerando os índices da figura 7.31, o *U-array* de tamanho 3×3×3 poderia ser visto como o apresentado na figura 7.32. Da mesma forma que na *U-matrix*, o cálculo dos valores *du*(*x*, *y*, *z*) pode ser feito tomando-se o valor médio ou a mediana dos elementos circunvizinhos.

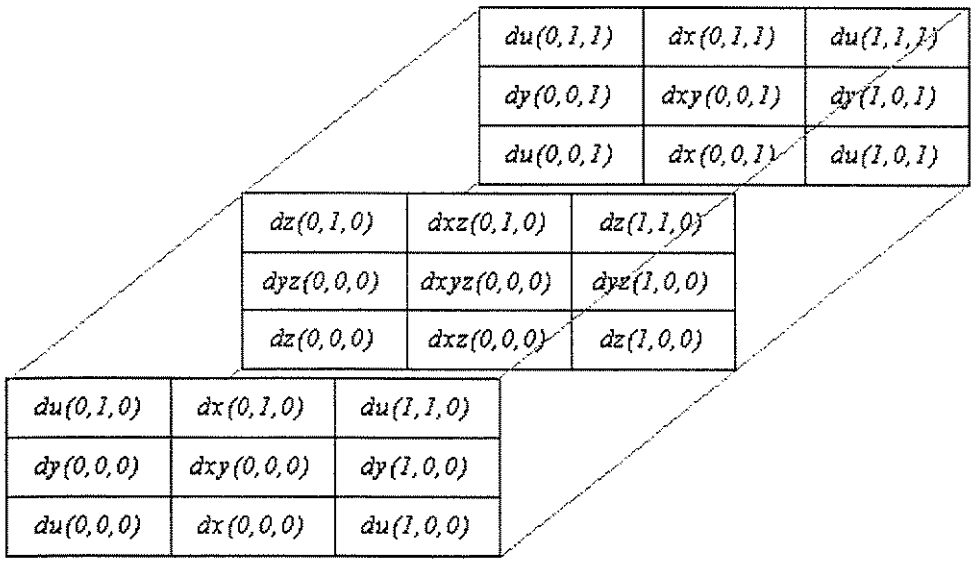
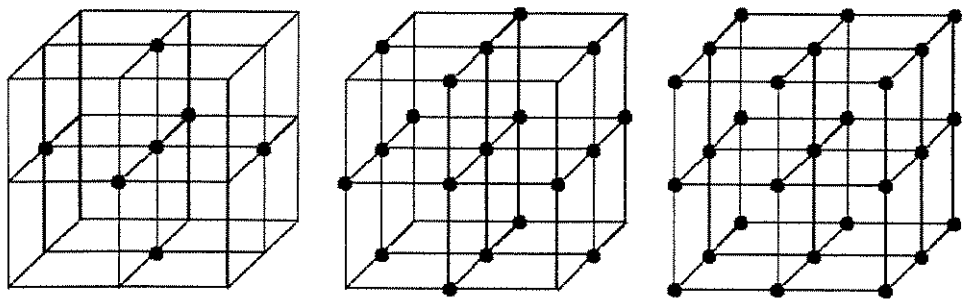


Figura 7.32 - Esquema do U-array para o SOM com tamanho 2×2×2 apresentado na figura 7.26.

7.4 Adaptação do algoritmo *SL-SOM* para mapas com dimensão maior que 2

Obviamente, a segmentação de um cubo ou hipercubo é mais complexa do que de uma imagem. No caso de um cubo, ou um mapa com topologia do espaço de saída tridimensional, o gráfico de regiões planas conectadas *versus* limiar (nível de cinza) torna-se um gráfico de um volume conectado *versus* o limiar.

A conectividade entre pixels torna-se conectividade entre voxels. O caso mais simples em 2D, i.e., conectividade 4-adjacente, pode ser estendida e computada de forma relativamente simples em uma imagem p-dimensional: para qualquer coordenada caso a distância entre dois pixels seja 1 eles estarão conectados. Exemplos de padrões de adjacência entre voxels, para o caso 3D, podem ser vistas na figura 7.33.



(a) 6-adjacente *(b) 18-adjacente* *(c) 26-adjacente*

Figura 7.33 - Exemplos de padrões de adjacência entre voxels (caso 3D)

A segmentação de uma imagem 3D pode também ser vista como a segmentação de uma seqüência de imagens 2D. O algoritmo *watershed* foi adaptado para efetuar tal operação (em 3D). Um dos principais problemas é a detecção dos marcadores, suavização da imagem e eliminação de voxels isolados. A operação RCC, rotulação de componentes (ou regiões) conectados, (Costa, 1996a), (Parker, 1994) passa a ser uma operação de rotulação de volumes conectados. Os tipos de conectividades utilizados neste trabalho foram os padrões 6 e 26-adjacente (ver figura 7.33).

No geral há um aumento de complexidade pois passa-se a trabalhar em maiores dimensões, porém em termos algorítmicos a filosofia é a mesma. Um efeito colateral é o tempo de processamento dos volumes que possuem bem mais voxels (pixels) do que as imagens. Novamente, isto constitui de um problema na atualidade, porém o autor considera que tais fatores, como o tempo de processamento e capacidade de armazenagem, serão atenuados à medida do avanço tecnológico na área da computação.

7.5 Exemplos de uso do *U-array* em mapas com dimensão maior que 2

Os dois exemplos mostrados a seguir são aplicações relativamente simples do *U-array*. O primeiro é a extensão da dimensão do espaço de saída de 2 para 3 do mapa utilizado na seção 5.5.3 e discutido no início deste capítulo, e o segundo exemplo usa a base de dados *chainlink* discutida na seção 5.5.1.

7.5.1 Mistura de Gaussianas no espaço $\Re^3$

Uma rede com topologia 8×8×8 foi utilizada (i.e., 512 neurônios), e o conjunto de dados é o apresentado na figura 5.58. Foram efetuadas 500 iterações do algoritmo de treinamento em lote (*batch*). A configuração dos neurônios obtida é apresentada na figura 7.34.

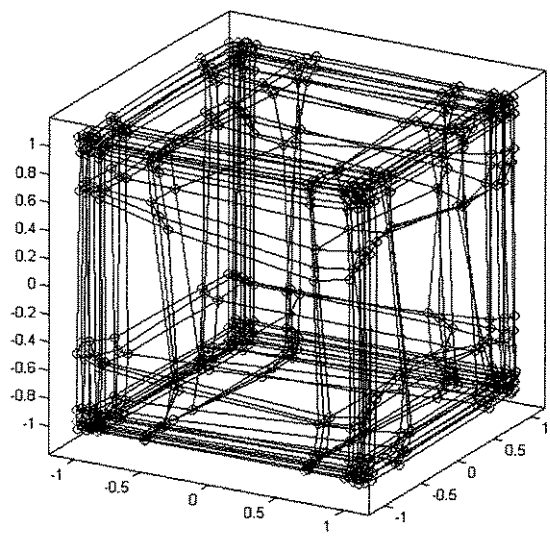


Figura 7.34 - Configuração dos neurônios após 500 épocas do algoritmo em lote.

O *U-array* para a figura 7.34 é apresentado na figura 7.35. Da mesma forma que na *U-matrix*, o tamanho de cada dimensão do *U-array* é $(2*N - 1)$, onde N corresponde a uma dada dimensão do mapa. Assim, temos o *U-array* com tamanho 15×15×15. Note que as distâncias entre neurônios foram capturadas pelo *U-array* e pelo fato de haver distâncias relativamente grandes entre neurônios pertencentes a cada agrupamento de dados ocorre a existência de bordas salientes entre os voxels que representam áreas distintas. Pelo fato do problema ser estruturado de forma que os agrupamentos estejam concentrados em vértices de um cubo, este resultado apresenta a simetria capturada automaticamente pelo SOM.

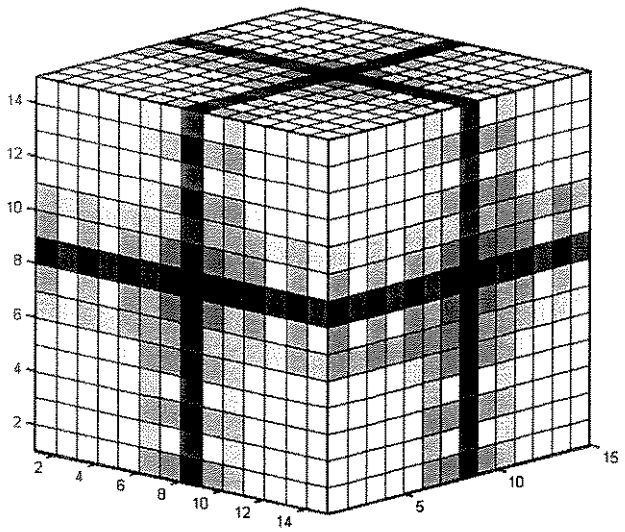


Figura 7.35 - U-array para a configuração dos neurônios apresentada na figura 7.34.

Uma outra forma de visualizar o *U-array* é efetuando-se cortes (*slices*) e visualizando-os como imagens bidimensionais. A figura 7.36 ilustra tais cortes efetuados sobre o *U-array* apresentado na figura 7.35. Os cortes foram efetuados de forma ortogonal ao eixo *z*. Da esquerda para a direita, de cima para baixo, o valor da coordenada *z* varia de 1 a 15. Note que houve um escalonamento linear junto às intensidades dos *voxels*, tanto na figura 7.35 como na figura 7.36. Esta faixa [0, 1] é posteriormente expandida para [0, 255] (figuras 7.37 e 7.38).

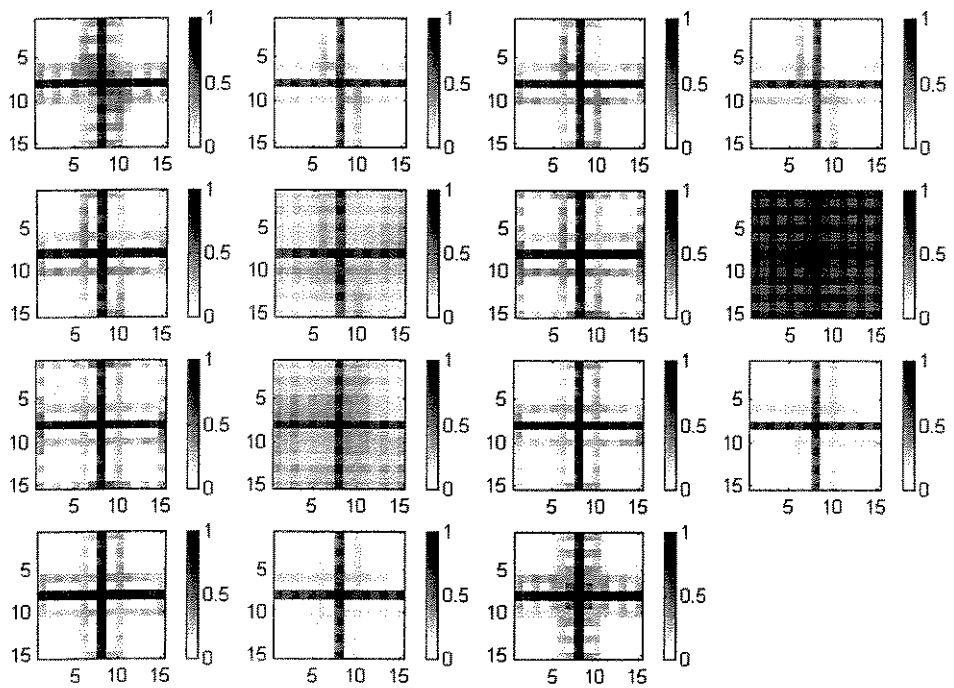


Figura 7.36 - Cortes ortogonais ao eixo z no U-array apresentado na figura 7.35.

Seguindo os passos do algoritmo SL-SOM, a partir da geração do *U-array* deve-se determinar os marcadores para a operação *watershed*. O gráfico apresentado na figura 7.37 ilustra o número de regiões (volumes) conectados do *U-array* à medida que elevamos o limiar (abcissa).

Note a existência de um platô (região de estabilidade) indicando a solução adequada em 8 marcadores. Porém, nota-se também que há um número elevado de volumes conectados. Efetuando uma filtragem, i.e., estabelecendo um limiar no qual apenas volumes significativos permaneçam no *U-array*, obtemos o gráfico apresentado na figura 7.38. Note que a solução em oito marcadores, que conduz a oito agrupamentos, que já era evidente na figura 7.37, fica explícita na figura 7.38. O valor limiar utilizado corresponde a 2.5% do total de voxels no *U-array* (3375). Assim, volumes conectados com menos que 85 voxels foram eliminados, i.e., tiveram seu rótulo igualado ao rótulo de fundo da imagem (*background*).

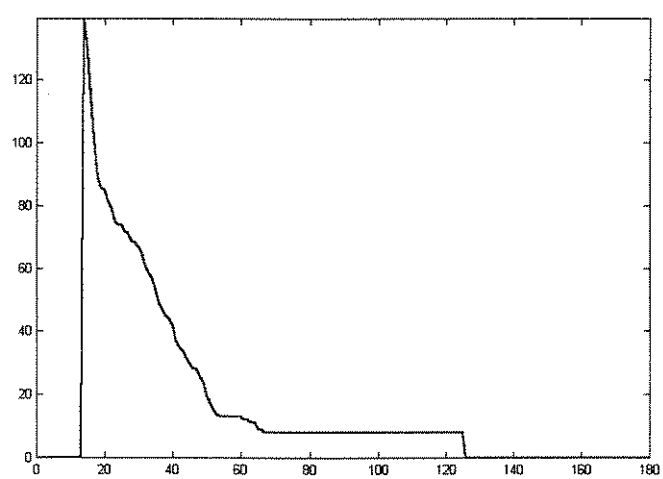


Figura 7.37 - Número de regiões (volumes) conectadas versus o valor do limiar do U-array.

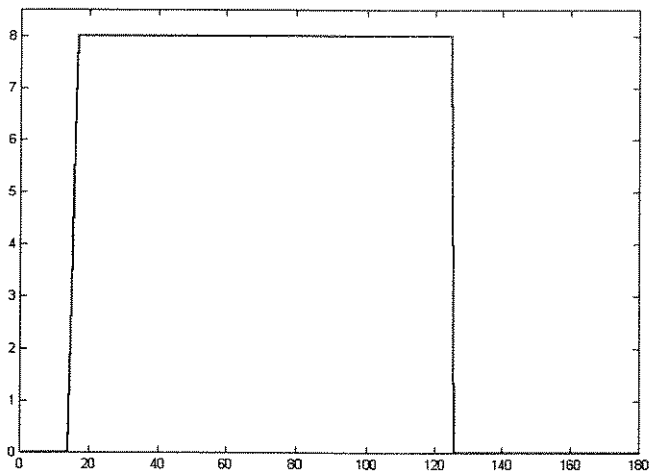


Figura 7.38 - Número de regiões (volumes) conectadas versus o valor do limiar do U-array, após eliminação de volumes menores que 2.5% do volume total do U-array.

Sobre o *U-array* resultante após eliminação dos pequenos volumes conectados, seguindo o algoritmo SL-SOM, utilizou-se o limiar 17 (veja figura 7.38) para efetuar binarização do *U-array*, resultando nos marcadores a serem utilizados no watershed. Tais marcadores são ilustrados nas figuras 7.39 e 7.40, na forma 3D e em cortes, respectivamente. O tipo de corte efetuado foi o mesmo descrito na figura 7.36.

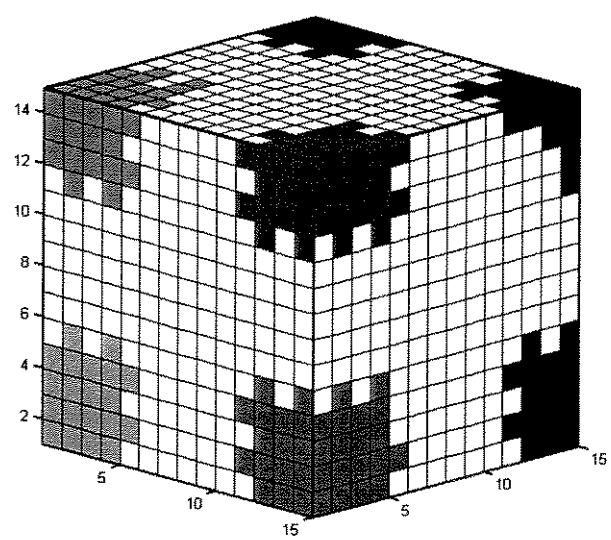


Figura 7.39 - Marcadores (volumes conectados) após binarização do U-array.

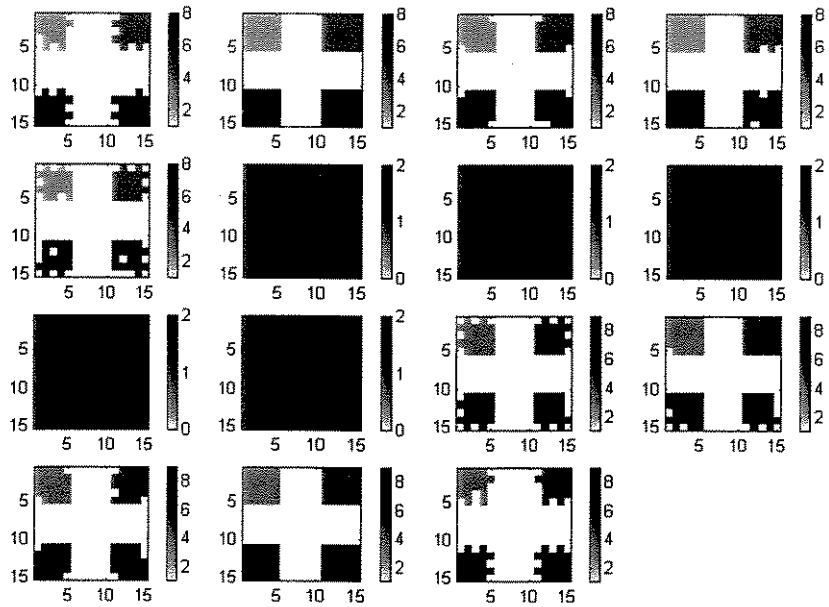


Figura 7.40 - Cortes ortogonais ao eixo z para o cubo contendo informações dos marcadores, apresentados na figura 7.39.

O resultado da operação *watershed* utilizando-se os marcadores (figuras 7.39 e 7.40) e o *U-array* (figuras 7.35 e 7.36) é apresentado na figura 7.41. As linhas de partição de águas (*watershed*) dividem o volume de forma similar à apresentada em outro problema com redes bidimensionais (ex. figura 5.8) e com este mesmo conjunto de dados (figura 5.66).

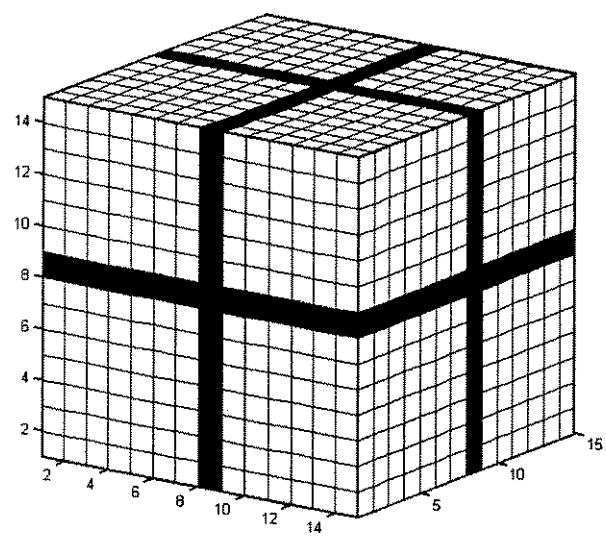


Figura 7.41 - Linhas da watershed obtidas utilizando os marcadores (fig. 7.39 e 7.40) e o U-array (figs. 7.35 e 7.36).

Da mesma forma que no caso bidimensional, os volumes conectados no *U-array* segmentado pela watershed são rotulados, i.e., atribui-se um código para cada volume conectado, e tais códigos são copiados para os neurônios correspondentes às posições no *U-array*. As figuras 7.42 e 7.43 ilustram o *U-array* rotulado, respectivamente a representação em 3D e os cortes ortogonais em relação ao eixo z do *U-array*. Note que voxels pertencentes a bordas (em preto na figura 7.41), inicialmente sem classe definida, foram classificados para as volumes rotulados pela regra vizinhos mais próximos.

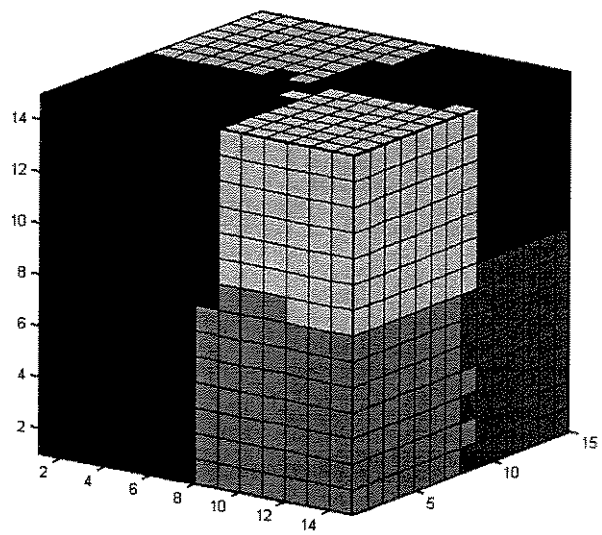


Figura 7.42 - U-array rotulado - representação em 3D.

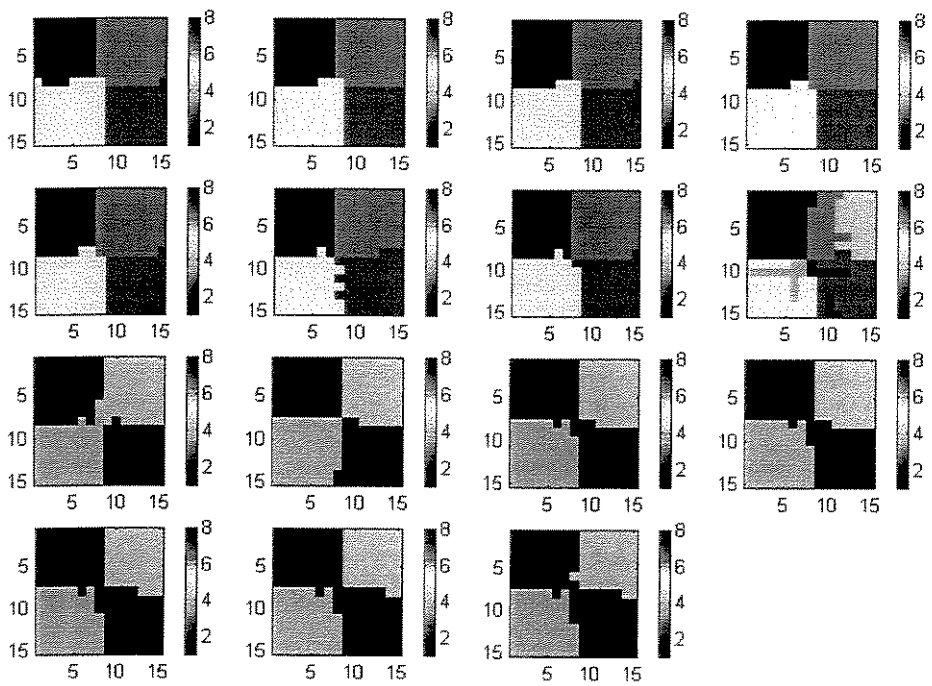


Figura 7.43 - U-array rotulado - representação por cortes ortogonais ao eixo z.

O mapa rotulado é apresentado nas figuras 7.44 a 7.46, respectivamente como volume, cortes ortogonais ao eixo z, e a configuração de neurônios, destacando diferentes cores para diferentes agrupamentos. Note que a classe de cada neurônio provêm da classe das posições relativas que os neurônios possuem no *U-array* rotulado.

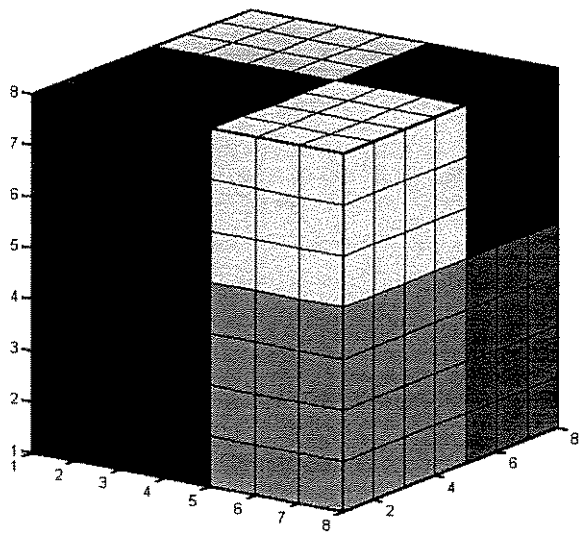


Figura 7.44 - Representação 3D do SOM com espaço de saída 8×8×8 particionado e rotulado.

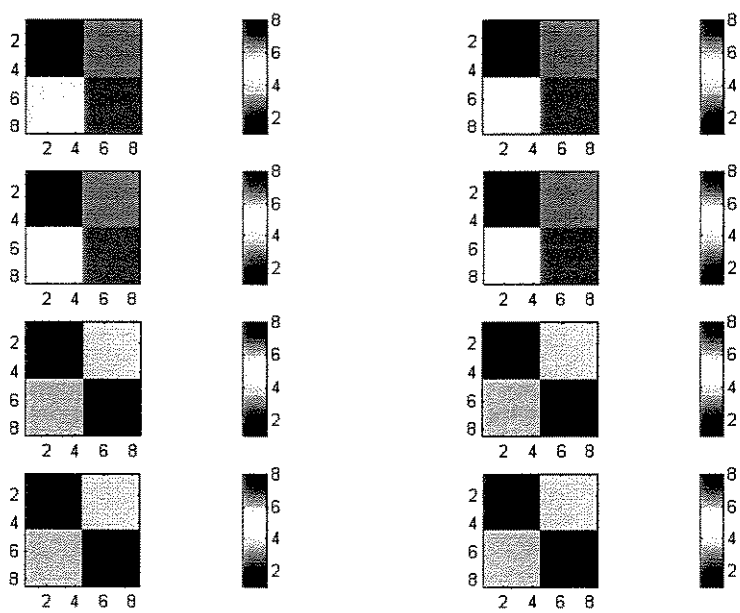


Figura 7.45 - Cortes ortogonais ao eixo z para o SOM rotulado e apresentado na figura 7.42.

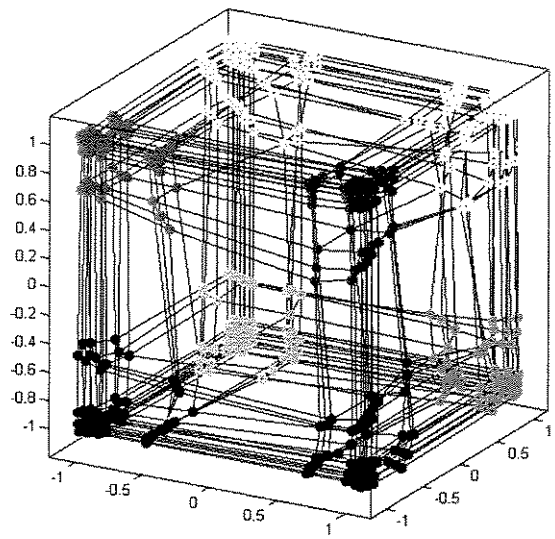


Figura 7.46 - Configuração de neurônios, destacando diferentes cores para diferentes agrupamentos.

A figura 7.47 ilustra a configuração de neurônios eliminando o efeito de neurônios inativos, i.e., $H(i, j, k) < 1$. Efetuando uma operação similar à mostrada na figura 5.26, onde cada neurônio é representado por uma esfera, vemos a configuração dos neurônios ativos e seus relacionamentos de vizinhança com os outros agrupamentos na figura 7.48. O raio da esfera utilizado na figura 7.48 foi 0.25.

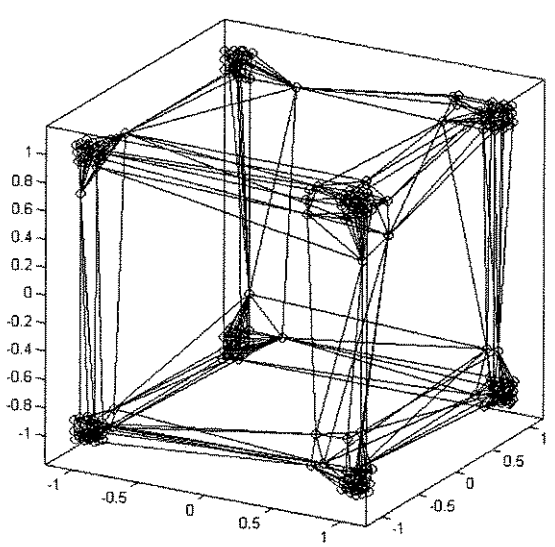


Figura 7.47: Configuração de neurônios eliminando o efeito de neurônios inativos, i.e., $H(i, j, k) < 1$.

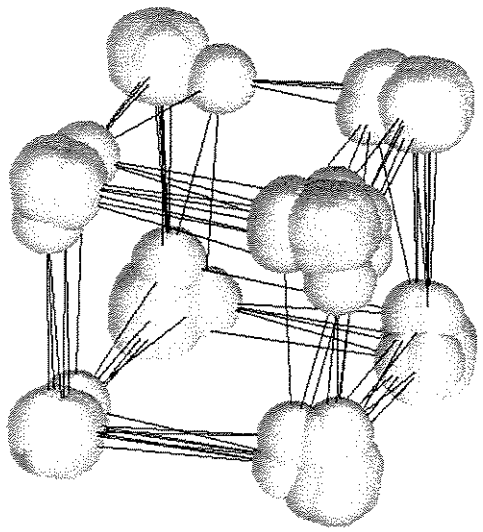


Figura 7.48: Neurônios ativos, H(i, j, k) > 0, e seus relacionamentos de vizinhança.

Note que sendo mais rigorosos em relação à permanência de neurônios em relação ao número de padrões que estão associados a eles, podemos eliminar o efeito de várias conexões e neurônios restando um imagem semelhante à apresentada na figura 7.2, porém neste caso, as classes vizinhas possuirão explicitamente relacionamentos de vizinhança.

A figura 7.49 ilustra os agrupamentos dos neurônios apresentados na figura 7.48 de forma rotulada. Note que cada neurônio possui influência igual no espaço (neste caso com raio 0.25 para razões de visualização), porém em termos que quantização de padrões vale a regra vizinhos mais próximos, e podemos pensar que a influência espacial de cada agrupamento cresce com o raio, a partir do centro de cada neurônio, até o infinito, a menos que duas regiões de influência se toquem em algum lugar no espaço, cessando o crescimento e gerando uma borda naquela posição. Igualmente ao que ocorreu no capítulo 5, todas as amostras foram classificadas corretamente, o que pode ser explicado pelo fato de que há uma forte estrutura de agrupamentos nos dados.

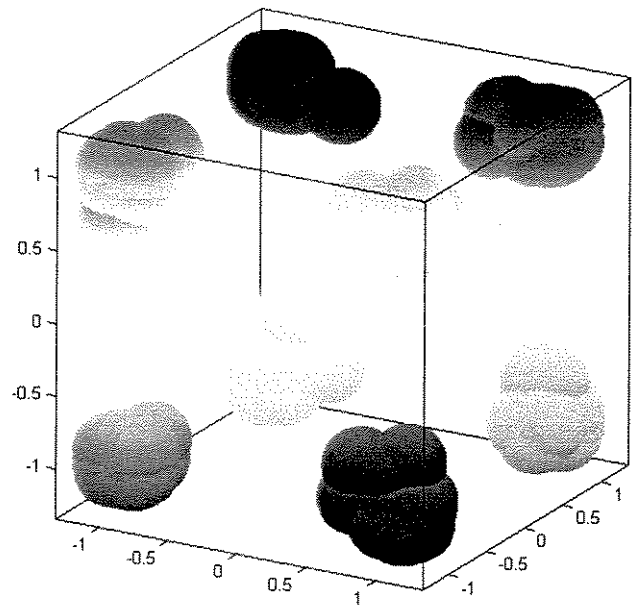


Figura 7.49: Agrupamentos de neurônios (ativos, H(i, j, k) > 1).

7.5.2 O conjunto de dados *chainlink*

Apesar do resultado em 2D ser suficiente para este problema, onde os dados estão no $\Re^3$, a motivação para o uso desta base com um SOM com espaço de saída 3D é ilustrativa, no sentido de tentar entender a adequação do mapa a problemas em que não há redução de dimensionalidade, e a topologia pode efetivamente ser preservada.

Igualmente à seção anterior, a estrutura do SOM utilizada no experimento continha 8×8×8 neurônios e o número de épocas (algoritmo batch) foi 200. A figura 7.50 ilustra a configuração da rede após inicialização linear, e a figura 7.51 ilustra a configuração obtida com o treinamento.

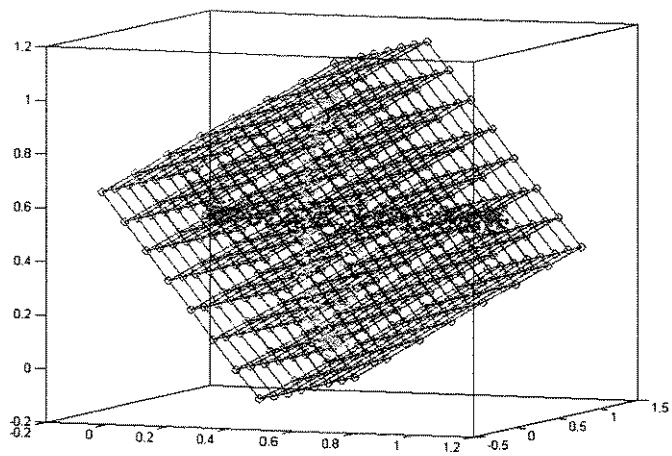


Figura 7.50: Configuração da rede após inicialização linear (dados também são mostrados).

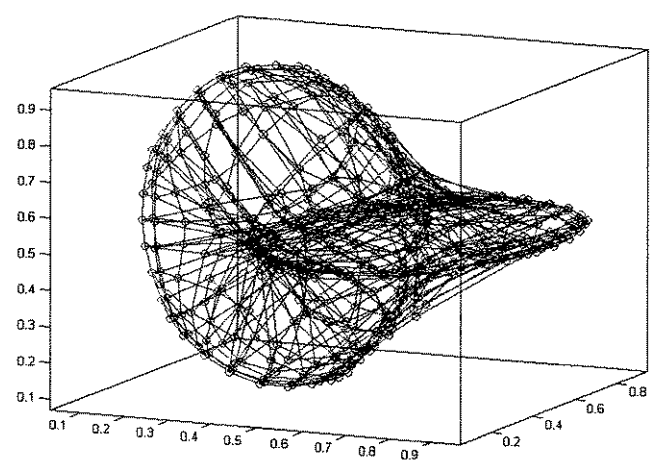


Figura 7.51: Configuração da rede após 200 épocas de treinamento (algoritmo batch).

O *U-array* relativo à rede apresentada na figura 7.51 é mostrado na figura 7.52, na forma de um cubo (3D), e na figura 7.53 na forma de cortes ortogonais ao eixo z. Em ambas as figuras, níveis de cinza escuros simbolizam dissimilaridade mais acentuada.

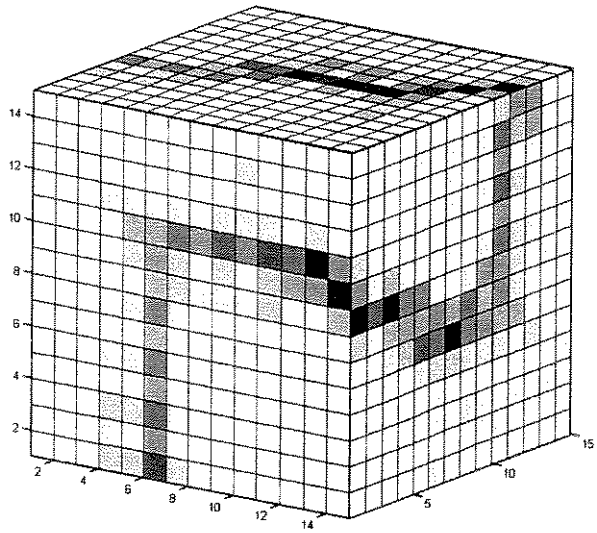


Figura 7.52: U-array relativo à rede apresentada na figura 7.51 (forma 3D)

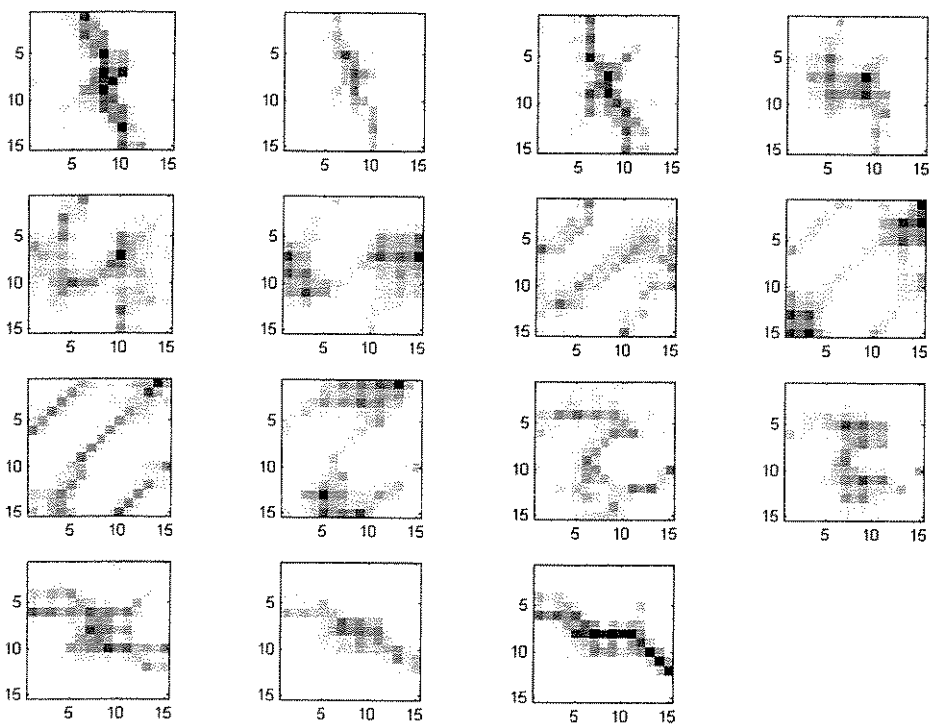


Figura 7.53: U-array relativo à rede apresentada na figura 7.51 (cortes ortogonais ao eixo z do U-array)

A figura 7.54 ilustra os marcadores utilizados para segmentação do *U-array*. Eles foram obtidos pela binarização do *U-array* utilizando o valor inicial (40) do platô de estabilidade no gráfico número de volumes conectados *versus* o limiar, cuja faixa foi de 40 a 132, estável para a solução dois agrupamentos. As linhas da watershed obtidas são mostradas nas figuras 7.55 e 7.56.

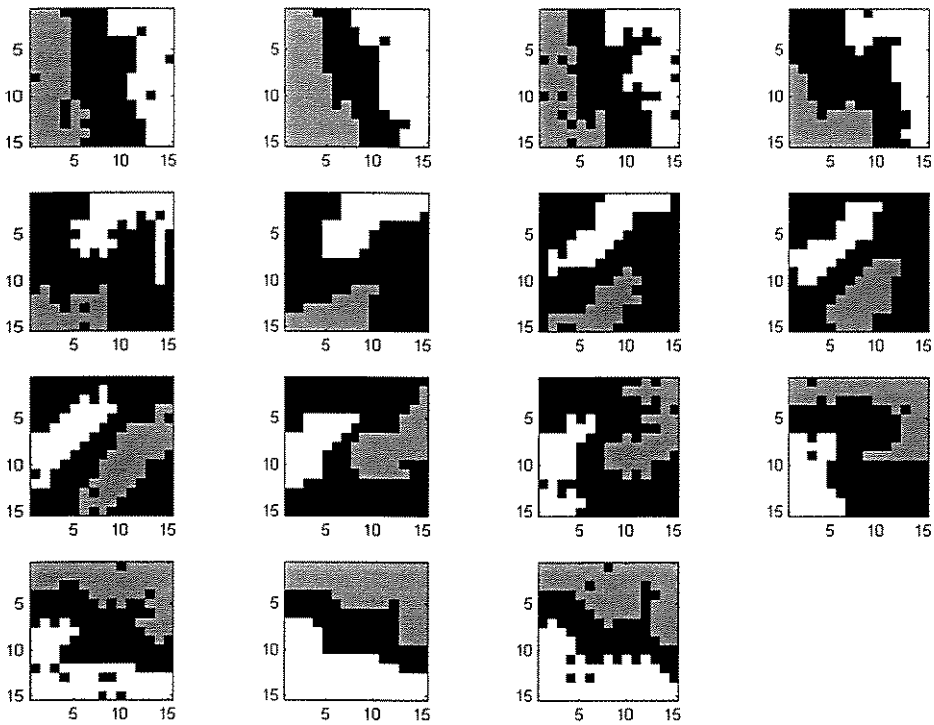


Figura 7.54: Marcadores utilizados para segmentação do U-array.

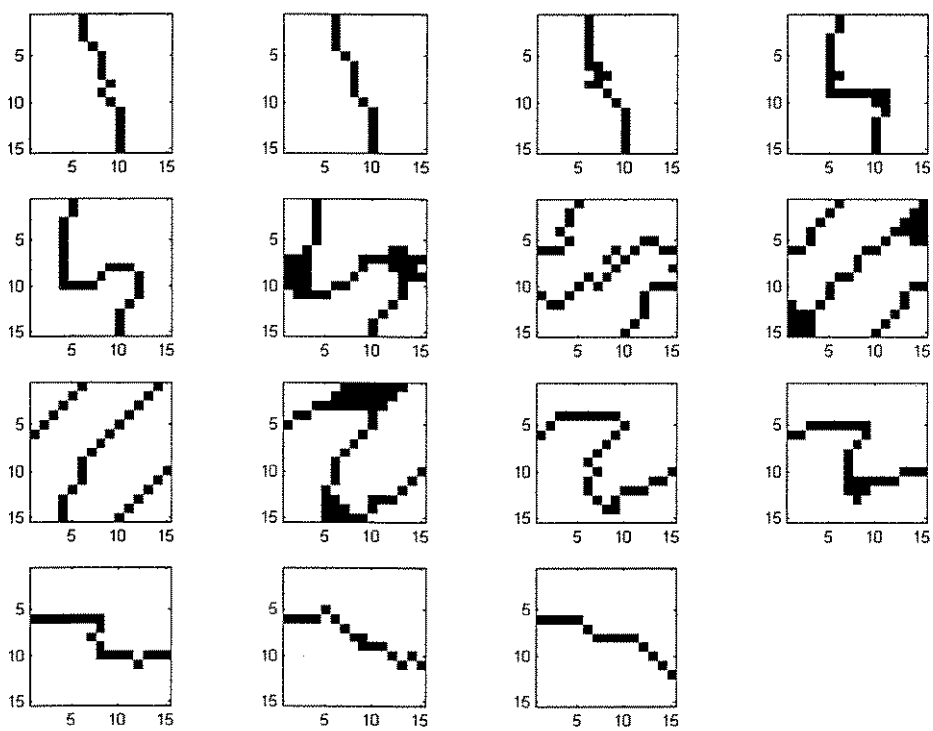


Figura 7.55: Linhas de watershed obtidas usando os marcadores mostrados na figura 7.54.

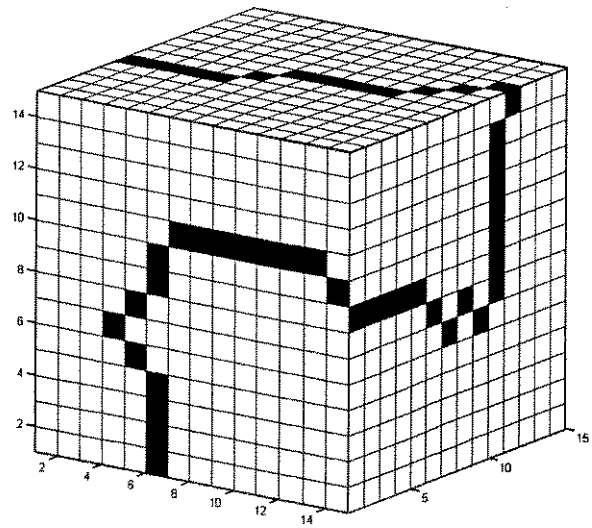


Figura 7.56: Linhas de watershed obtidas usando os marcadores mostrados na figura 7.54 (representação em 3D).

O *U-array* rotulado (duas regiões) é apresentado nas figuras 7.57 e 7.58.

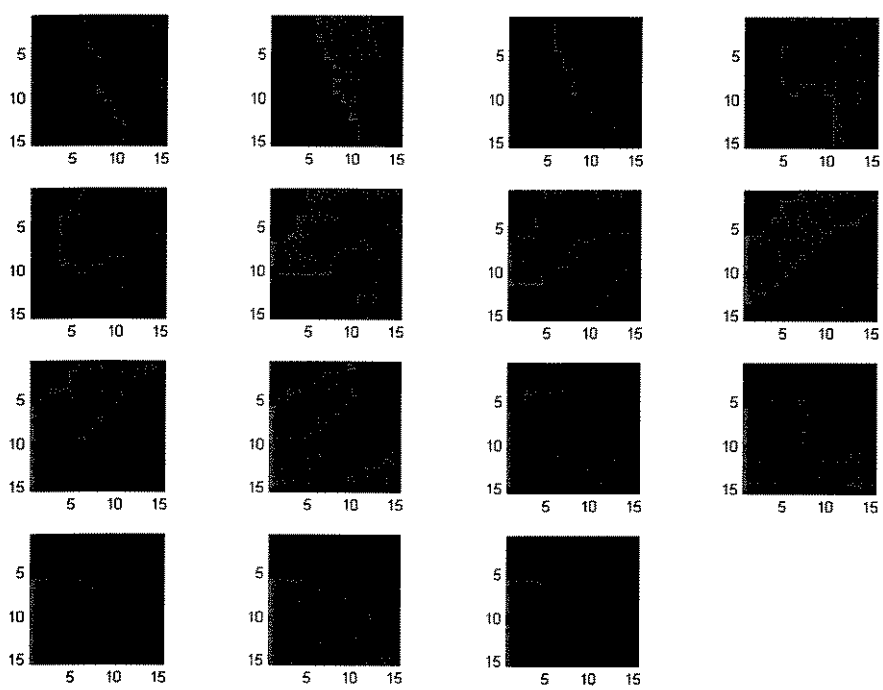


Figura 7.57: U-array rotulado (duas regiões) - cortes ortogonais ao eixo z do U-array.

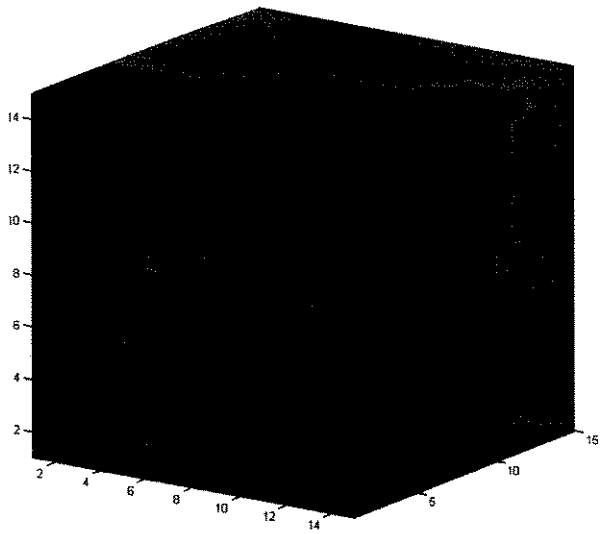


Figura 7.58: U-array rotulado (duas regiões) - representação 3D.

Da mesma forma que discutido na seção anterior, os rótulos obtidos na segmentação do *U-array* são copiados para as posições respectivas do mapa de Kohonen. Desta forma, obtemos as figuras 7.59 e 7.60.

A figura 7.61 ilustra a configuração dos neurônios eliminando o efeito dos neurônios inativos, enquanto que a figura 7.62 ilustra a influência espacial dos agrupamentos detectados, limitando o raio das esferas em 0.05. Nesta última figura, buscou-se um ângulo de visualização que enfoca a solução obtida pelo método para este conjunto de dados não linearmente separável. Relações entre neurônios vizinhos também são mostradas.

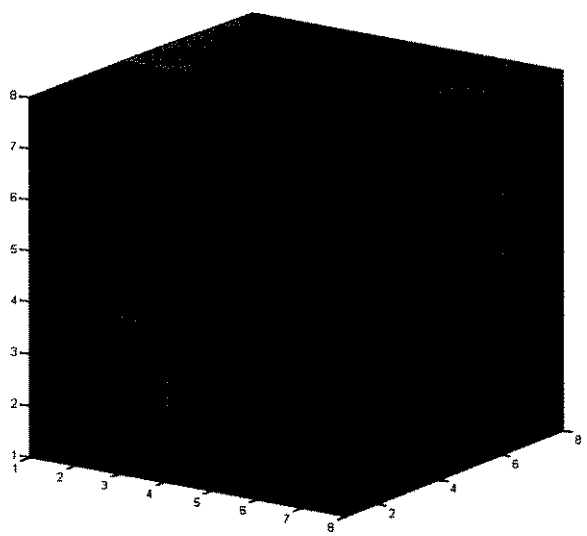


Figura 7.59: Espaço de saída da rede SOM rotulada (duas regiões) - representação 3D.

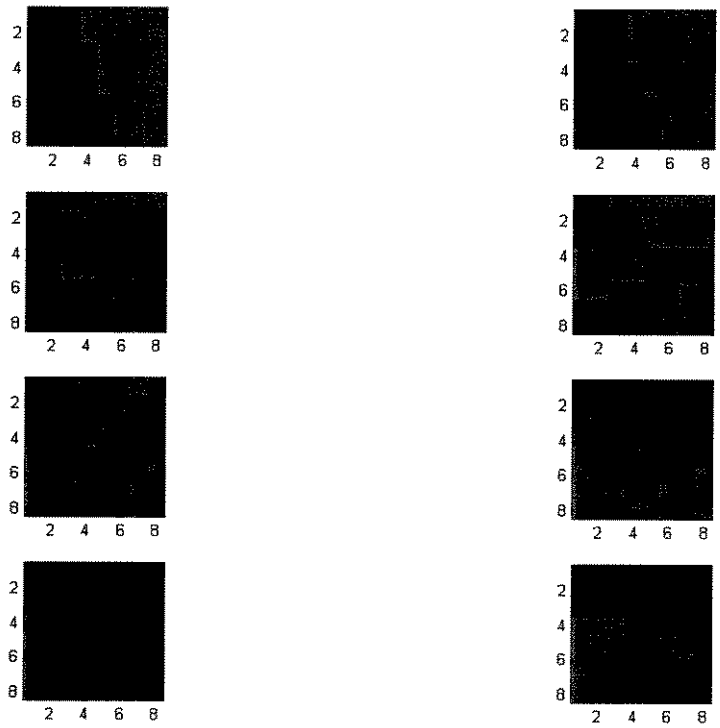


Figura 7.60: Espaço de saída da rede SOM rotulada (duas regiões): representação por cortes ortogonais ao eixo z.

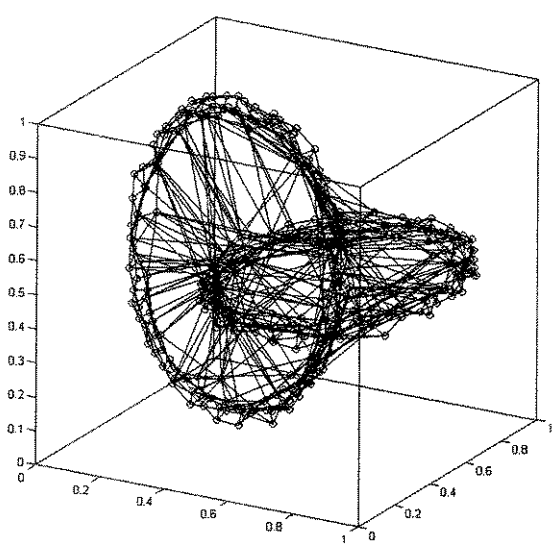


Figura 7.61: Configuração dos neurônios eliminando o efeito dos neurônios inativos

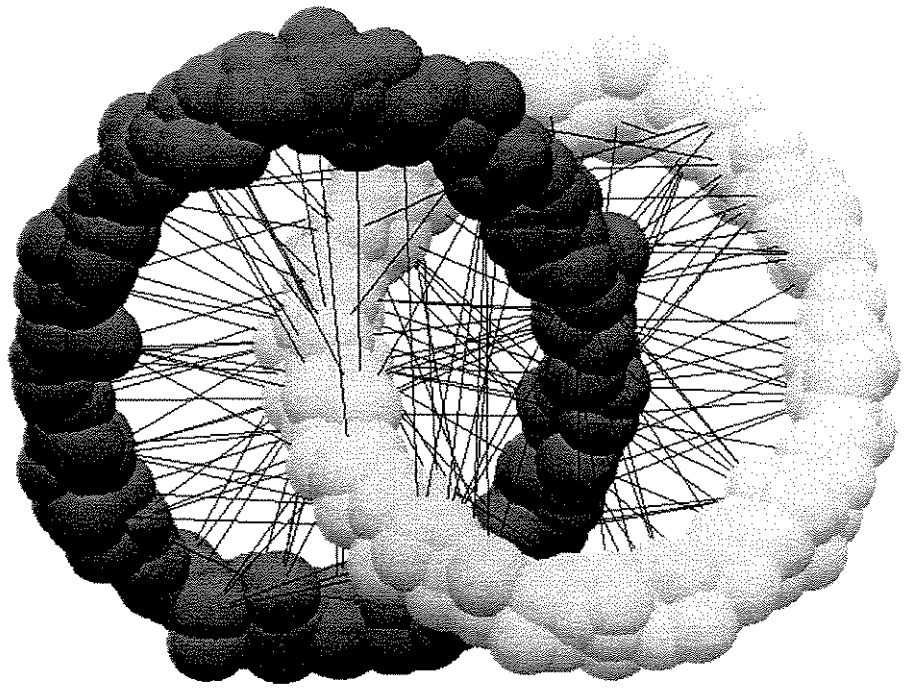
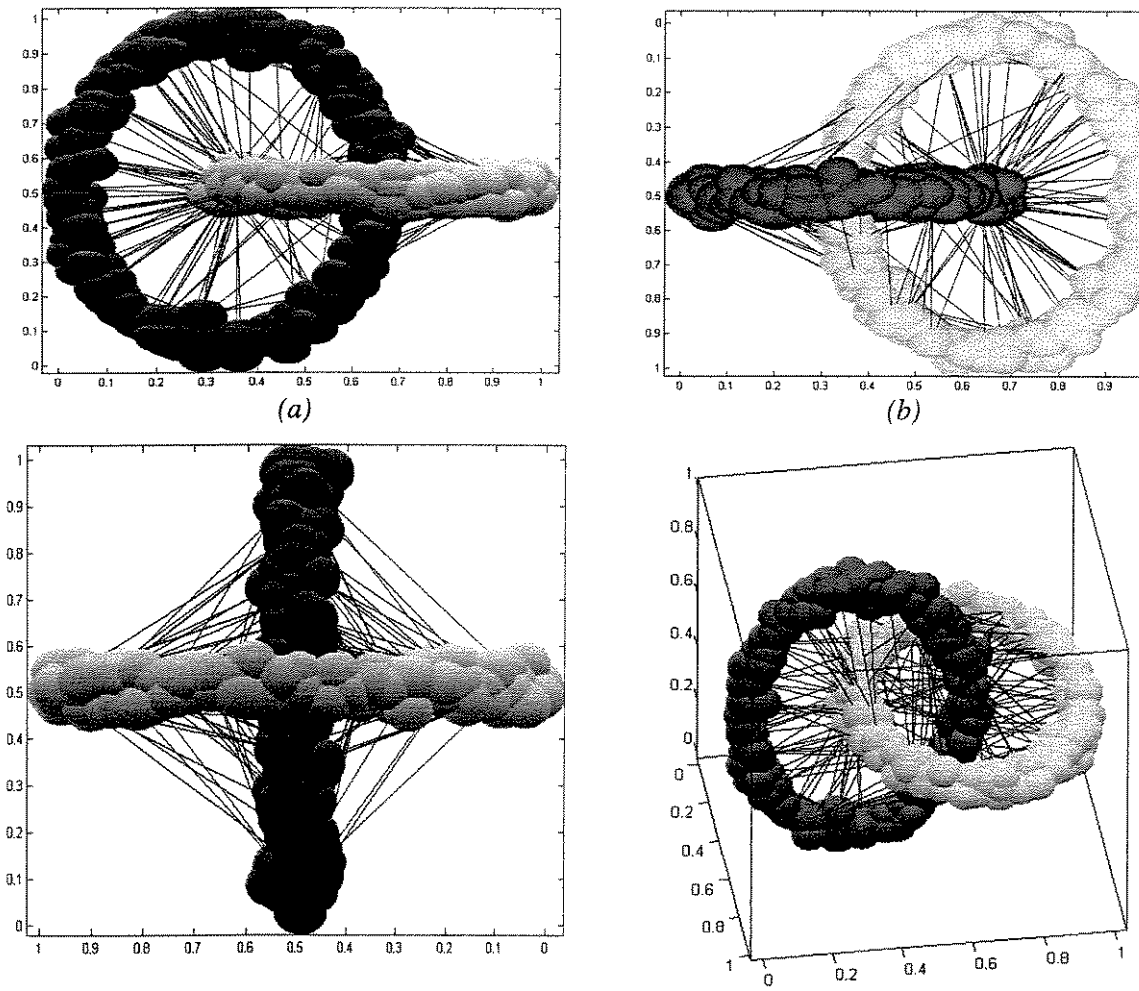


Figura 7.62: Configuração dos neurônios eliminando o efeito dos neurônios inativos. Raio de influência para neurônios: 0.05.

As figuras 7.63 (a-d) ilustram outros ângulos da figura 7.62. Note que ambos os agrupamentos de neurônios fecham os círculos relacionados aos agrupamentos originais dos dados. No caso bidimensional, a deformação da rede 2D apenas simula tal fechamento.

A figura 7.64 ilustra os agrupamentos para um raio de influência $r = 0.15$, para cada neurônio ativo. À medida que r aumenta podemos visualizar no espaço as regiões de influência de cada agrupamento. O sistema desenvolvido permite ao usuário visualizar uma seqüência de passos entre uma faixa de valores $[r_{min}, r_{max}]$, permitindo, da mesma forma que a superfície de influências faz para o caso bidimensional (ver capítulo 3), a compreensão da estrutura obtida utilizando-se ferramentas gráficas e de simulação numérica.



(a) *(b)*

(c) *(d)*

Figura 7.63 (a-d): ilustrações de outros ângulos da estrutura de agrupamentos detectada

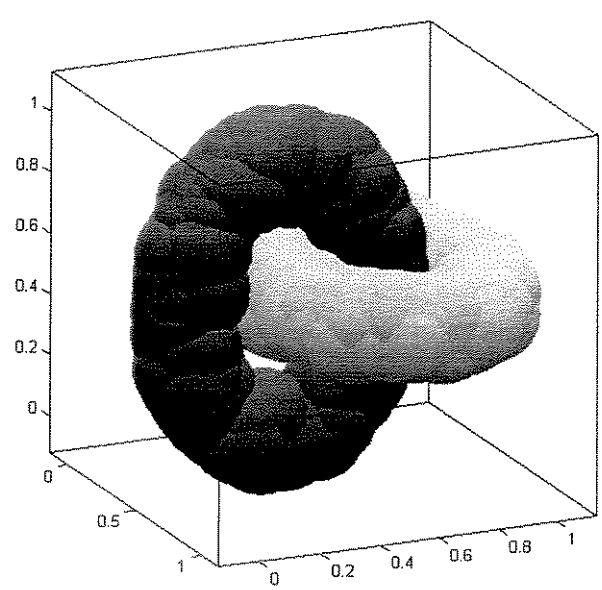


Figura 7.64: Agrupamentos de neurônios para um raio de influência r = 0.15.

7.6 Sumário

A principal contribuição deste capítulo foi mostrar que os métodos desenvolvidos no capítulo 5 e 6 podem ser estendidos para redes com espaços de saída com dimensão maior que 2. Atualmente grande parte das aplicações e do uso dos mapas de Kohonen usam espaços de saída bidimensionais porque a principal função é a visualização do mapeamento do espaço original (de elevada dimensão) no *display* ou conjunto de neurônios da rede.

Porém, geralmente há perda de relações topológicas entre mapeamentos de espaços com dimensão mais elevada para espaços com menor dimensão, e muitas vezes tal informação pode ser bastante útil. Um exemplo ilustrado na seção 7.1 mostra que, em um problema relativamente simples, classes vizinhas no espaço $\Re^3$ foram mapeadas em posições relativamente distantes no mapa 2D. Em aplicações de mineração de dados e descoberta de conhecimento em bases de dados, pode ser importante saber não só a existência das classes e seus elementos componentes mas também que classes são semelhantes. Esta última informação motivou o desenvolvimento do *U-array* e da extensão dos métodos para sua segmentação e análise.

Discutimos também brevemente sobre índices que podem mensurar erros topográficos. Trabalhos futuros podem incluir tais medidas como indicadores para ajuste da dimensão do espaço de saída do SOM, de forma que tenhamos algoritmos de treinamento que adaptem automaticamente a dimensão da rede para um dado problema, como sugerido por Bauer & Villmann (1997). As ferramentas de análise do SOM, descritas neste capítulo, podem ser estendidas de forma similar para 4 ou mais dimensões. Como a visualização torna-se bastante problemática para mapas de dimensão maior que 3, podemos apenas colher resultados sintéticos sobre a estrutura de agrupamentos detectada pelo SL-SOM, na forma de relatórios indicando os agrupamentos e seus relacionamentos.

Capítulo 8

Conclusões

Tratamos de um problema bastante complexo, classificação automática de padrões, ou análise de agrupamentos, no qual partições devem ser automaticamente encontradas para conjuntos de dados onde cada objeto é descrito por p-variáveis. Pesquisadores das mais diversas áreas têm publicado inúmeras aplicações e novos algoritmos, porém, não há, na atualidade, métodos gerais que sejam adaptáveis a uma vasta gama de tipos de dados e geometrias dos agrupamentos.

O método advogado nesta tese usa conjuntos de protótipos para representar agrupamentos, ou classes de objetos, o que resulta em uma capacidade do sistema de se adaptar a diferentes geometrias no espaço dos dados. A rede de Kohonen, que é o tipo de rede neural auto-organizável mais importante na atualidade, é utilizada tanto no aspecto relacionado à quantização vetorial, i.e., efetuar uma aproximação da densidade de probabilidade do conjunto de dados, como também em relação às propriedades topológicas entre padrões e suas classes.

Foram discutidos aspectos relacionados à auto-organização, análise de agrupamentos e medidas de similaridade, e a rede de Kohonen, SOM, no capítulo 3, onde discutimos sobre sua estrutura e seu processo de treinamento. A superfície de influências foi definida como uma forma de visualização do processo de quantização, útil quando o espaço de entrada possui dimensão 2. Abordamos aspectos relacionados à classificação usando o SOM, e apresentamos métodos de eliminar a influência de neurônios inativos na configuração obtida via treinamento.

O uso de morfologia matemática para segmentar a *U-matrix* de mapas treinados foi descrito no capítulo 5, resultando em um algoritmo que efetua particionamento e rotulação automática, o *SL-SOM*. O algoritmo *watershed*, que funciona como uma combinação de métodos baseados em detecção de bordas e crescimento de regiões, foi aplicado com sucesso à segmentação da *U-matrix*, após escolha de marcadores apropriados, encontrados após uma análise de estabilidade das regiões conectadas da *U-matrix* para vários níveis de cinza da imagem. Aspectos relacionados à segmentação foram discutidos e exemplos mostraram características do processo. Em particular, comparações foram feitas com os

métodos *k-means* e as misturas de densidades de probabilidades, com parâmetros estimados pelo algoritmo EM. Os resultados com o SOM, particionado pelos métodos propostos, mostraram que, mesmo sem conhecimento *a priori* do número de classes ou da forma da densidade dos agrupamentos, resultaram em soluções adequadas, compatíveis com os métodos estatísticos (que usam informação privilegiada do número de classes). Em casos como o do *chainlink*, métodos estatísticos são incapazes de descobrir a estrutura inerente dos agrupamentos, sendo um problema linearmente não separável. Tal solução foi efetivamente resolvida pelo SOM, e apresentada nos capítulos 5 e 6. Idéias de como melhorar contrastes entre vales e bordas na *U-matrix* foram também descritas no capítulo 5.

A extensão do método SL-SOM foi motivada pela possibilidade de detectar subclasses de dados nas regiões particionadas inicialmente. O algoritmo *TS-SL-SOM*, descrito no capítulo 6, propõe uma estrutura hierárquica, uma árvore de mapas, a qual possui dinâmica de geração orientada pelos dados. Regiões dão origem a novos mapas, e caso estes não possuam duas ou mais sub-regiões, são podados da estrutura. Cada sub-mapa é treinado com parte do conjunto de dados do mapa pai: apenas os padrões que são quantizados na região do mapa pai que deu origem a tal sub-mapa. Este processo pode ser visto como um particionamento recursivo da base de dados, onde em cada estágio há focalização de atenção nos subconjuntos de dados sendo separados. Desse modo, a árvore resultante, explica relacionamentos hierárquicos entre dados e classes.

O *U-array* foi proposto como extensão da *U-matrix*, aplicado em mapas com espaço de saída com dimensão maior que 2, cuja motivação principal é a manutenção da topologia dos dados e das classes, que geralmente é perdida quando mapeamos um espaço de dimensão mais elevada em um espaço de saída de menor dimensão. Exemplos e resultados foram apresentados e comentados. Uma breve revisão de medidas de erros topográficos em mapas neurais foi efetuada, e tais medidas podem ser úteis no projeto de métodos de treinamento que possam adaptar automaticamente a dimensionalidade do espaço de saída de acordo com as características do banco de dados.

Muitos autores ainda utilizam o SOM como ferramenta de agrupamentos de dados de forma bastante similar ao *k-means*, i.e., escolhem o número de neurônios igual ao número esperado de classes, por exemplo Balakrishnan *et al.* (1994), Bezdek & Pal (1995), Mangiameli *et al.* (1996), Murtagh (1996), Waller *et al.* (1998) e Flexer (1997, 1999). O SOM é mais que o *k-means*, pois incorpora informações topológicas entre os neurônios representantes dos subgrupos de dados, que pode inclusive ser utilizado para diversos fins, como por exemplo, redução da distorção de imagens após compressão e transmissão em canais com ruído (Kangas & Kohonen, 1996). Problemas relacionados a erros no mapeamento topográfico têm sido bastante estudados, ver por exemplo Bauer *et al.* (1999),

e algoritmos que incorporam funções para minimizar tais erros, online, começam a ser apresentados (Kirk & Zurada, 1999).

Em aplicações tais como os de mineração de dados e descoberta de conhecimento em bases de dados, os algoritmos apresentados podem ser bastante úteis, apresentando não só os grupos existentes, mas também seus elementos e relações. Tais operações, como a partição do mapa de Kohonen, ainda atualmente feitas de forma manual, inclusive em softwares comerciais, como Clementine (1998), e pelo WEBSOM (Kohonen, 1998), podem ser automatizados pelo uso do SL-SOM, e suas extensões. Os avanços conseguidos nesta tese contrastam com os métodos convencionais do uso do SOM como ferramenta apenas de visualização de dados, como mostrados, por exemplo, em Timmins *et al.* (1999).

Pela facilidade de acesso a grandes volumes de informações, que temos atualmente, e que suponhamos aumentar cada dia mais, ferramentas como as descritas nesta tese podem, em um futuro não distante, serem partes em softwares de análise de dados, complexas, com visualização gráfica e suporte por sistemas especialistas, ou mesmo softwares comuns, como planilhas eletrônicas.

8.1 Possíveis extensões deste trabalho

Classificação de padrões são tarefas fundamentais em nosso cotidiano e na ciência. A aplicação das técnicas apresentadas nesta tese podem ser estendida para qualquer tipo de processo que colete ou possua informações, e que necessite redução de dimensionalidade ou sumarizações de grandes volumes de dados. Aplicações comuns em engenharia incluem problemas de processamento de sinais e imagens, reconhecimento de fala, segmentação, filtragem e compressão de imagens, esquemas alternativos de organização e recuperação de informações em bancos de dados, algoritmos para percepção e identificação de padrões por computadores, entre outras.

Muito menos que um fim em si, este trabalho abre novas portas para várias futuras investigações sejam teóricas ou práticas.

O relaxamento do critério de pertinência onde neurônios e padrões pertençam a apenas uma classe para partições *fuzzy* é uma extensão natural do trabalho. Porém, critérios devem ser estabelecidos caso desejemos criar sub-mapas filhos de partições *fuzzy*.

A análise dos mapas descrita nesta tese supõe o treinamento tenha sido efetuado com sucesso. O uso do algoritmo em lote previne contra variabilidades decorrentes da seqüência de apresentação dos padrões na rede, e a inicialização linear configura a grade de neurônios em uma posição privilegiada em relação à inicialização aleatória. Apesar de existir apenas prova de convergência do SOM em casos bastante simples, espera-se que tenhamos em um futuro não longe métodos que assegurem convergência no treinamento do SOM, de forma mais independente das escolhas feitas no início do treinamento.

A dinâmica de crescimento da árvore de mapas (algoritmo TS-SL-SOM) pode ser otimizada, estudando-se mais profundamente características estatísticas dos subconjuntos de dados quantizados em cada região detectada nos mapas. Outras regras para determinação de parâmetros ideais para cada mapa / sub-mapa são esperadas em breve. Um exemplo imediato é a adaptação da topologia do espaço de saída das redes, seguindo índices como os apresentados em Bauer *et al.* (1999), de forma que minimizemos o erro topográfico no mapeamento de um espaço com dimensão mais elevada que o espaço de saída do mapa. Espera-se que algoritmos genéticos possam também ser aplicados na busca de soluções adequadas de parâmetros para um determinado problema, como por exemplo, dimensões da rede.

Espera-se obter maior profundidade na formalização do processo de agrupamento de neurônios obtido pelo modelo SL-SOM. O capítulo 5 brevemente descreve a segmentação pela *watershed* como um método similar ao método de agrupamento aglomerativo por ligações simples (LS), caso restrinjamos as fusões dos neurônios de forma restrita aos elementos circunvizinhos. Propriedades matemáticas de tal método devem ser pesquisadas com maior profundidade. O uso de outros métodos que não o *watershed* no SL-SOM têm sido recentemente desenvolvidas, como é o caso de algoritmos de particionamento de grafos. Sendo o espaço de saída do SOM um grafo conectado com propriedades especiais, tais métodos podem tornar a tarefa de segmentação mais simples e econômica, em termos computacionais, principalmente quando usamos mapas com espaços de saída de dimensão elevada.

A estrutura hierárquica gerada pelo TS-SL-SOM pode ser aplicada como uma interface de acesso e visualização de bases de dados convencionais, organização de informações contendo textos, sons e imagens, e até mesmo para informações contidas na internet. Aplicações futuras incluem a possibilidade de implementação de tal interface com a linguagem VRML, de forma que seja possível ao usuário efetuar sua busca na base de dados navegando virtualmente em imagens de *U-matrizes* estruturadas na forma de árvores. Desta forma, poder-se-á navegar em direção ao resultado da busca, indo na direção de contextos mais próximos ao desejado a cada avanço no nível da árvore. A possibilidade de

estruturar grandes volumes de dados pelo uso do SOM, como por exemplos, bibliotecas virtuais, têm motivado vários pesquisadores e empresas de tecnologia de bancos de dados e inteligência computacional, porém as ferramentas desenvolvidas até então ainda requerem intervenções manuais, veja por exemplo Merkl (1998).

Referências Bibliográficas

Adams, R., Butchart, K., & Davey, N. (1999). Hierarchical classification with a competitive evolutionary neural tree. *Neural Networks*, 12, 541-551.

Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P. (1998). *Automatic subspace clustering of high dimensional data for data mining applications.* Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 94-105, Seattle, WA.

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In: B. N. Petrov & F. Csaki (eds.), *Proc. of the Second International Symposium on Information Theory*, Academia Kiado, Budapest, p. 267-281.

Akaike, H. (1974). A new look in statistical model identification. *IEEE Trans. Automatic Control*, v. AC-19, no. 6, pp. 716-722.

Allinson, N. M. (1992). Self-organising neural maps and their applications. In J. G. Taylor and C. L. T. Mannion, editors, *Theory and Applications of Neural Networks*, pages 101-120. Springer, London, UK.

Ambros-Ingerson, J., Granger, R., & Lynch, G. (1990). Simulation of paleo-cortex performs hierarchical clustering. *Science*, vol. 247, pp. 1344-1348.

Anderberg, M.R. (1973), *Cluster Analysis for Applications*, New York: Academic Press, Inc.

Arabie, P., Hubert, L. J., De Soete, G. (Eds) (1996). *Clustering and Classification.* World Scientific Pub Co: Singapore.

Ashby, W. R. (1962), Principles of the self-organizing system. In: Von Foester, H. & Zopf, Jr., G.W. (org.), *Principles of Self-Organization.* Oxford: Pergamon, pp. 255-278. (Reimpresso em Ashby, W.R. (1981), *Mechanisms of Intelligence.* Seaside (USA): Intersystems, pp. 65-89).

Avanzolini, G., Barbini, P., Gnudi, G., Grossi, A. (1991). Cluster analysis of clinical data measured in the surgical intensive care unit. *Computer Methods and Programs in Biomedicine*, vol. 35, no. 3, pp. 157-170.

Balakrishnan, P.V.S., Cooper, M., Jacob, V., & Lewis, P. (1994). A study of the classification capabilities of neural networks using unsupervised learning: A comparison with K-means clustering. *Psychometrika*, vol. 59, pp. 509-525.

Balakrishnan, P.V.S., Cooper, M., Jacob, V., & Lewis, P. (1996). Comparative performance of the FSCL neural net and K-means algorithm for market segmentation. *European J. of Operational Research*, vol. 93, pp. 346-357.

Ball, G. & Hall, D. (1967). A clustering technique for summarizing multivariate data. *Behavior Science*, vol. 12, pp. 153-155.

Ball, N. R. (1994). Reinforcement learning in Kohonen feature maps. In Maria Marinaro and Pietro G. Morasso, editors, *Proc. ICANN'94, Int. Conf. on Artificial Neural Networks*, volume I, pages 663-666, London, UK, 1994. Springer.

Banfield, J.D. and Raftery, A.E. (1993). Model-Based Gaussian and Non-Gaussian Clustering, *Biometrics*, 49, 803-821.

Baraldi, A. & Parmiggiani, F. (1995). A self-organizing neural network merging Kohonen's and ART models. In *Proc. ICNN'95, IEEE Int. Conf. on Neural Networks*, volume V, pages 2444-2449, Piscataway, NJ.

Barnard, E. & Casasent, D. (1990). Shift invariance and the neocognitron. *Neural Networks*, v. 3, p. 403-410.

Barrera, J., Banon, G.J.F., and Lotufo, R. A. (1994). Mathematical Morphology Toolbox for the KHOROS System. *In: Proc. of the Conf. on Image Algebra and Morphological Image Processing V*, Intl. Symposium on Optics, Imaging and Instrumentation, SPIE's Annual Meeting. San Diego, CA.

Barrera, J., Banon, G., Lotufo, R. & Hirata Jr., R.(1997). MMach: a Mathematical Morphology Toolbox for the KHOROS System. Tech. Report RT-MAC-9704. IME / University of São Paulo.

Bauer, H.-U. & Pawelzik, K. (1992). Quantifying the Neighborhood Preservation of Self-Organizing Feature Maps. *IEEE Trans. on Neural Networks*, vol. 3, no. 4, pp. 570-579.

Bauer, H. U. (1994). Oriented ocular dominance bands in the Self-Organizing Feature Map. In: Maria Marinaro and Pietro G. Morasso, (eds.), *Proc. ICANN'94, Int. Conf. on Artificial Neural Networks*, vol. I, pp. 42-45, London, UK.

Bauer, H.-U., & Villmann, T. (1997). Growing a Hypercubical Output Space in a Self-Organizing Feature Map. *IEEE Trans. on Neural Networks,* vol. 8, no. 2, pp. 226-233.

Bauer, H.-U., Herrmann, M., & Villmann, T. (1999). Neural maps and topographic vector quantization. *Neural Networks*, June.

Beale, R. & Jackson, T. (1990). *Neural Computing: An Introduction.* Adam Hilger: Bristol, UK.

Benaim, M., Fort, J. C., & Pages, G. (1997). Almost sure convergence of the one-dimensional Kohonen algorithm. In M. Verleysen, editor, *5th European Symposium on Artificial Neural Networks* ESANN '97. Proceedings, pages 193-8. D facto, Brussels, Belgium, 1997.

Ben-Arieh, D., e Triantaphyllou, E. (1992). Quantifying data for group technology with weighted fuzzy features. International Journal of Production Research, v. 30 n. 6, pp. 1285-1299

Bensmail, H., Celeux, G., Raftery, A.E., and Robert, C.P. (1997), Inference in model-based cluster analysis, *Statistics and Computing*, 7, 1-10.

Beucher, S.& Lantuéjoul, C. (1979). Use of Watersheds in Contour Detection. *Proc. Int'l Workshop Image Processing, Real-Time Edge and Motion Detection / Estimation*, Rennes, France.

Bezdek, J.C. (1981), *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York: Plenum Press.

Bezdek, J.C. & Pal, S.K. (1992), *Fuzzy Models for Pattern Recognition: Methods That Search for Structures in Data*. New York: IEEE Press.

Bezdek, J. & Pal, N. (1993a). Fuzzification of the self-organizing feature map: will it work? *Proceedings of the SPIE--The International Society for Optical Engineering*, 2061:142-62.

Bezdek, J. & N. R. Pal. (1993b). An index of topological preservation and its application to self-organizing feature maps. *In Proc. IJCNN-93, Int. Joint Conf. on Neural Networks,* Nagoya, volume III, pages 2435-2440, Piscataway, NJ.

Bezdek, J. & N. R. Pal. (1995a) A note on self-organizing semantic maps. *IEEE Transactions on Neural Networks*, 6(5):1029-1036.

Bezdek, J. & N. R. Pal. (1995b). An index of topological preservation for feature extraction. *Pattern Recognition*, 28(3):381-91.

Bezdek, J. C., Tsao, E. C. K. & N. R. Pal (1992). Fuzzy Kohonen clustering networks. In *Proc. IEEE Int. Conf. on Fuzzy Systems*, pages 1035-1043, Piscataway, NJ, 1992.

Binder, D.A. (1978), Bayesian Cluster Analysis, *Biometrika*, 65, 31-38.

Binder, D.A. (1981), Approximations to Bayesian Clustering Rules, *Biometrika*, 68, 275-285.

Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*, Oxford: Oxford University Press.

Blackmore J., Miikkulainen R. (1993). Incremental Grid Growing: Encoding High-Dimensional Structure into a Two-Dimensional Feature Map In: *Proceedings of the IEEE International Conference on Neural Networks* (ICNN'93), San Francisco, CA, USA.

Blackmore J., Miikkulainen R. (1995). Visualizing High-Dimensional Structure with the Incremental Grid Growing Neural Network, In: Prieditis A., Russel S. (eds.) *Machine Learning: Proceedings of the 12th International Conference*, USA.

Blashfield, R.K. and Aldenderfer, M.S. (1978), "The Literature on Cluster Analysis, *Multivariate Behavioral Research*, 13, 271-295.

Bock, H. H. (1974), *Automatische Klassification*. Vandenhoeck & Rupprecht, Göttingen.

Bock, H. H. (1985), On Some Significance Tests in Cluster Analysis, *Journal of Classification*, 2, 77-108.

Boehme, H. J., Braumann, U. D., & Gross, H. M. (1994). A neural network architecture for sensory controlled internal simulation. In: Maria Marinaro and Pietro G. Morasso, (eds.), *Proc. ICANN'94, Int. Conf. on Artificial Neural Networks*, vol. II, pp. 1189-1192

Bortolan, G., Degani, R., e Willems, J. L. (1992). ECG classification with neural networks and cluster analysis. In: *Proceedings of the 18th IEEE Annual Conference on Computers in Cardiology*, p. 177-180. Los Alamitos, CA

Bouguettaya, A. (1996). On-line clustering. *IEEE Trans. on Knowledge and Data Engineering*, v. 8, n. 2, pp. 333-339.

Bouguettaya, A., e Le Viet, Q. (1998). Data clustering in a multidimensional space. *Information Sciences*.

Bouton, C., Cottrell, M., Fort, J. C., & Pagès, G. (1991). Self-organization and convergence of the Kohonen algorithm. In N. Bouleau and D. Talay, editors, *Probabilités Numériques*, chapter V. 2, pp. 163-180. INRIA, Paris, France.

Bouton, C. & Pagès, G. (1994). Convergence in distribution of the one-dimensional Kohonen algorithms when the stimuli are not uniform. *Advances in Applied Probability*, 26:80-103.

Bozdogan, H. (1987). Model selection and Akaikes's information criterion (AIC): General theory and its analytical extensions. *Psychometrica*, v. 52, no. 3, p. 345-370.

Bozdogan, H. (1992). Mixture-model cluster analysis and choosing the number of clusters using a new information complexity ICOMP, AIC, and MDL model-selection criteria. In: Bozdogan, H. (ed.), *Multivatiate Statistical Modelling*, Vol. II, Kluwer Academic Publishers, Holland, Dordrecht.

Bozdogan, H. (1993). Choosing the number of component clusters in the mixture-model using a new information complexity criterion of the inverse-Fisher information matrix. In: O. Opitz et al. (eds.) *Studies in classification, data analysis, and knowledge organization*, p. 40-54. Springer-Verlag, Heidelberg.

Brodley, C.E., & Utgoff, P. E. (1995). Multivariate decision trees. *Machine Learning*, vol. 19, pp. 45-77.

Buhmann, J., Lades, M., & von der Malsburg, C. (1990). Size and distortion invariant object recognition by hierarchical graph matching. In: *Proc. of the Intl. Joint Conference on Neural Networks*. p. 411-416.

Bullock, T., Orkand, R., e Grinnell, A. (1977). *Introduction to Nervous Systems*. Freeman, San Francisco.

Burel, G. & Pottier, I. (1991). Vector quantization of images using Kohonen algorithm. Theory and implementation. *Revue Technique Thomson-CSF*, 23(1):137-159.

Cagnoni, S., Coppini, G., Livi, R., e Valli, G. (1991). Knowledge-based system for the diagnosis and treatment of hypertension. Journal of Biomedical Engineering, v. 13 n. 2, pp. 119-125.

Calinski, T. and Harabasz, J. (1974), A Dendrite Method for Cluster Analysis, *Communications in Statistics*, 3, 1-27.

Canny, J. (1986). A Computational Approach to Edge Detection, *IEEE Trans. Pattern Analysis and Mach. Intelligence*, vol. 8, pp. 679-698.

Carpenter, G. & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Comp. Vision, Graphics and Image Proc.*, vol. 37, pp. 54-115.

Carpenter, G. & Grossberg, S. (1988). The Art of Adaptive Pattern Recognition by a Self-Organizing Neural Network. *IEEE Computer*, March, pp. 77-88.

Carpenter, G. & Grossberg, S. (1991). *Pattern Recognition by Self-Organizing Neural Networks*. Cambridge, MA: MIT Press.

Choi, D.-I. & Park, S.-H. (1994). Self-Creating and Organizing Neural Networks. *IEEE Trans. on Neural Networks*, vol. 5, no. 4, pp. 561-575.

Chao, T-H. & Stoner, W.W. (1993). Optical implementation of a feature-based neural network with application to automatic target recognition. *Applied Optics*, v. 32, n. 8, pp. 1359-1369.

Clementine User Guide, Integral Solutions Limited, 1998.

Cole, A. J. (1969). *Numerical Taxonomy*. New York: Academic Press.

Cormack, R. M. (1971). A review of classification. *Journal of the Royal Statistical Society*, Series A, vol. 134, pp. 321-367.

Corral, J.A., Guerrero, M., & Zufiria, P. (1994). Image compression via optimal vector quantization: A comparison between SOM, LBQ and K-means algorithms. *In Proc. ICNN'94, Int. Conf. on Neural Networks*, pages 4113-4118, Piscataway, NJ.

Costa, J.A.F. (1996a). Sistema de reconhecimento de padrões visuais invariante a transformações geométricas utilizando redes neurais artificiais de múltiplas camadas. Dissertação de mestrado. Universidade de São Paulo, S. Carlos, SP. Janeiro. 171 páginas.

Costa, J. A. F. (1996b). Sistemas de Visão Computacional: Conceitos e Exemplo de Aplicação em Classificação Automática de Objetos. In: *Anais do II Congresso Internacional de Engenharia Industrial*. Piracicaba, SP. Outubro.

Costa, J. A. F. (1996c). "PSR Invariant Object Recognition in Noise Corrupted Images Using Neural Networks". In: *Proceedings of the Third International Congress of Electronic Engineering* (INTERCON'96). IEEE: Trujillo, Perú.

Costa, J. A. F., e Gonzaga, A. (1996a). A System for Invariant Visual Pattern Recognition Using Multilayer Feedforward Neural Networks". In: *Proceedings of the 7th International Conference on Signal Processing, Applications & Technology* (ICSPAT'96), vol. 2, pp. 1148-1152, Boston, MA, USA: Miller Freeman.

Costa, J. A. F., e Gonzaga, A. (1996b). Sistema de reconhecimento de objetos para inspeção visual automática em ambientes industriais. In: Anais do *IEEE INDUSCON 1996 Conferência sobre aplicação industrial da energia elétrica)*, IEEE e IEEE Industry Applications Society. S. Paulo, Agosto de 1996. Páginas 1 a 8, vol. "Automação e Controle".

Costa, J.A.F., & Netto, M. L. A. (1997). Parts classification in assembly lines using multilayer feedforward neural networks, *Proc. of the 1997 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 3872-3877. Orlando, FL, October 12-15.

Costa, J.A.F., Netto, M.L.A., & Gonzaga, A. (1997a). Mechanical parts classification in assembly lines using artificial neural networks, In: *Contrôle Qualité par Vision Artificielle* 1997 (ISBN 2.85428.450.X) pp. 356-361. Cépaduès-Éditions: Toulouse – France.

Costa, J.A.F., Mascarenhas, N. & Netto, M.A (1997b). Cell nuclei segmentation in noisy images using morphological watersheds. In: *Applications of Digital Image Processing XX*. A. Tescher (Ed.). *Proc. of the SPIE*, vol. 3164, pp. 314-324.

Costa, J.A.F., & Netto, M. L. A. (1998). An Approach for Estimating the Number of Clusters in Multivariate Data by Self-Organizing Maps. In*: Anais do V Simpósio Brasileiro de Redes Neurais (SBRN'98)*, p. 33-38. Dezembro, Belo Horizonte, MG.

Costa, J.A.F., & Netto, M. L. A. (1999a). Estimating the Number of Clusters in Multivariate Data by Self-Organizing Maps. *International Journal of Neural Systems.* Vol. 9, no. 3, pp. 195-202.

Costa, J.A.F. & Netto, M.A. (1999b). Cluster analysis using self-organizing maps and image processing techniques. *Proc. of the 1999 IEEE Intl. Conf. on Systems, Man, and Cybernetics*, Tokyo, Japan.

Costa, J.A.F. & Netto, M.L.A. (1999c). Automatic Data Classification by a Hierarchy of Self-Organizing Maps, *Proc. 1999 IEEE Intl. Conf. on Systems, Man, and Cybernetics*, Tokyo, Japan.

Cottrell, M., Fort, J. C., & Pagès, G. (1994*). Two or three things that we know about the Kohonen algorithm.* Technical Report 31, Université Paris 1, Paris, France.

Cottrell, M., Fort, J. C., & Pagès, G. (1995). Comment about 'analysis of the convergence properties of topology preserving neural networks'. *IEEE Trans. on Neural Networks*, 6(3):797-799, 1995.

Cottrell, M. (1997).. Theoretical aspects of the SOM algorithm. *In Proceedings of WSOM'97, Workshop on Self-Organizing Maps*, Espoo, Finland, June 4-6, pages 246--267. Helsinki University of Technology, Neural Networks Research Centre, Espoo, Finland, 1997.

Cottrell, M., Fort, J.C., Pagès, G. (1998). Theoretical aspects of the SOM algorithm, *Neurocomputing* (21)1-3 pp. 119-138.

Darken, C. & Moody, J. (1991). Note on learning rate schedules for stochastic otimization. In: *Advances in Neural Networks Information Processing Systems*, vol. 3. Morgan Kauffmann, Los Altos, CA.

Day, W. H. & Edelsbrunner, H. (1984). Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, vol. 1, pp. 7-24.

Debrun, M. (1996). A idéia da auto-organização. In: Debrun et al. (1996) Eds. *Auto-Organização: Estudos Interdisciplinares*. Pp. 3-24. CLE, Unicamp. Campinas, SP.

Debrun, M., Gonzales, M.E.Q., & Pessoa Jr., O. (1996). Auto-Organização: Estudos Interdisciplinares. CLE, Unicamp. Campinas, SP.

Dedieu, E. & Mazer, E. (1992). An approach to sensorimotor relevance. In F. J. Varela and P. Bourgine, (eds.), Toward a Practice of Autonomous Systems. *Proc. First European Conf. on Artificial Life*, pp. 88-95, Cambridge, MA. MIT Press.

Dempster, A.P., Laird, N.M., & Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc.*, B., vol. 39, pp. 1-38.

Diday, E., and Simon, J. C. (1978). Clustering analysis. In: Fu, K.S. (ed.) *Communications and Cybernetics 10: Digital Pattern Recognition*, pp. 47-94. Berlin: Springer-Verlag.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, vol. 1, pp. 269-271.

Dony, R. & Haykin, S. (1995). Neural network approaches to image compression. *Proc. of the IEEE*, 83(2):288-303.

Dubes, R. and Jain, A. K. (1980). Clustering Methodologies in Exploratory Data Analysis, in *Advances in Computers*, vol. 19, M. Yovits (ed.), Academic Press, 1980, pp. 113-228.

Duda, R.O. and Hart, P.E. (1973), *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons.

Duda, R.O., Hart, P.E. & Stork, D. G. (1998). Pattern Classification (2nd Ed.). New York: Wiley.

Duran, B.S. and Odell, P.L. (1974), *Cluster Analysis*, New York: Springer-Verlag.

Durbin, R., Miall, C., & Mitchison, G. (eds.) (1989). *The Computing Neuron*. Reading, MA: Addison-Wesley.

Erwin, E., Obermayer, K., & Schulten, K. (1992a). Self-organizing maps: Stationary states, metastability and convergence rate. *Biological Cybernetics*, 67(1):35-45, 1992.

Erwin, E., Obermayer, K., & Schulten, K. (1992b). Self-organizing maps: Ordering, convergence properties and energy functions. *Biological Cybernetics*, 67(1):47-5.

Everitt, B. S. (1974). *Cluster Analysis*. Heinemann Education Books, London.

Everitt, B.S. (1981), A Monte Carlo investigation of the likelihood ratio test for the number of components in a mixture of normal distributions, *Multivariate Behavioral Research*, 16, 171-80.

Everitt, B.S. (1993). *Cluster Analysis*. 3rd Ed., New York: Wiley.

Everitt, B.S. and Hand, D.J. (1981), *Finite Mixture Distributions*, New York: Chapman and Hall.

Facon, J. (1996). *Morfologia matemática: Teoria e exemplos*. Editora Univ. Champagnat, PUC-PR. Curitiba.

Favata, F. & Walker, R. (1991). A study of the application of Kohonen-type neural networks to the Travelling Salesman Problem. *Biological Cybernetics*, 64(6):463-468.

Ferrán, E. A. & Ferrara, P. (1992). Clustering proteins into families using artificial neural networks. *Computer Applications in the Biosciences*, 8(1):39-44.

Firmin, C., Hamad, D., Postaire, J., & Zhang, R. (1997). Gaussian neural networks for glass bottles inspection: A learning procedure. *International Journal of Neural Systems*, v. 8, n. 1, pp. 41-46.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Ann. Eugen.*, vol. 7, pp. 179-188.

Flanagan, J. A. (1994). *Self-Organizing Neural Networks*. PhD thesis, École Polytechnique Fédérale de Lausanne, Lausanne.

Flanagan, J.A. (1996). Self-organization in Kohonen's SOM. *Neural Networks*, 9:1185-1197

Flexer, A. (1997). Limitations of Self-organizing Maps for Vector Quantization and Multidimensional Scaling, in Mozer M.C., et al.(eds.), *Advances in Neural Information Processing Systems* 9, MIT Press/Bradford Books, pp.445-451.

Flexer, A (1999). On the use of self-organizing maps for clustering and visualization. In: *Proceedings of the 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases* (PKDD-99), Prague, Springer-Verlag.

Florek, K., Luckaszewiez, J., Perkal, J., Steinhaus, H., & Zubrzchi, S. (1951). Sur la liason et la division des points d'un ensemble fini. *Colloquium Methematicum*, 2, pp. 282-285.

Fort, J. C. & Pages, G. (1996). About the Kohonen algorithm: strong or weak self-organization? *Neural Networks*, 9(5):773-85.

Frank, T., Kraiss, K.-F. & Kuhlen, T. (1998). Comparative analysis of fuzzy ART and ART-2A network clustering perform-ance. *IEEE Trans. on Neural Networks*, v. 9, 544-559.

Freeman, W. J. (1991). The physiology of perception. *Scientific American*, vol. 264, pp. 78-85.

Friedman, H. P. & Rubin, J. (1967). On some invariant criteria for grouping data. *J. Amer. Statist. Assoc.*, vol. 62, pp. 1159-1178.

Fritzke B. (1993) Kohonen Feature Maps and Growing Cell Structures - a Performance Comparison, *Advances in Neural Information Processing Systems* 5, C.L. Gibs, S. J. Hanson, J. D. Cowan (eds.), Morgan Kaufmann, San Mateo, CA, USA

Fritzke B. (1994) Growing Cell Structures - A Self-Organizing Network for Supervised and Unsupervised Learning, *Neural Networks*, Vol. 7, No. 9, p.p.1441 - 1460.

Fritzke B. (1995). A Growing Neural Gas Network Learns Topologies. In: G. Tesauro, D. S. Touretzky and T. K. Leen, (eds.), *Advances in Neural Information Processing Systems 7*, MIT Press, Cambridge MA, p. 625-632.

Fritzke, B. (1996). Growing Self-organizing Networks - Why?. In: M. Verleysen, (ed.), *ESANN'96: European Symposium on Artificial Neural Networks*, D-Facto Publishers, Brussels, pp. 61-72.

Fukushima, K. (1975). Cognitron: a self-organizing multilayered neural network. *Biological Cybernetics*, 20:121 - 136.

Fukushima, K. (1980). Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, *Biological Cybernetics*, v. 36, n. 4, p. 193-202.

Fukushima, K., Miyake, S., & Ito, T. (1983). Neocognitron: a neural networks model of visual pattern recognition. *IEEE Trans. Systems, Man & Cybernetics*, vol. SMC-13, pp. 826-834.

Fukushima, K. (1988). A neural network model for visual pattern recognition. *Computer*, v. 21, n. 3, pp. 65-75.

Fukushima, K. (1991). Neural network models for visual pattern recognition. *IEICE Transactions*, v. E74, n. 1, p. 179-190.

Fukushima, K. (1993). Improved Generalization Ability Using Constrained Neural Network Architectures. In: *Intl. Joint Conference on Neural Networks*, Proceedings. Nagoya, p. 2049-2054.

Gasteiger, J. & Zupan, J. (1993). Neural networks in chemistry. Angewandte Chemie, Intrenational Edition in English, 32(4):503-527.

Ghosh-Roy, R., Habiballah, I.O., Stonham, T.J., & Irving, M.R. (1998). On-line legal aid: Markov chain model for efficient retrieval of legal documents. *Image and Vision Computing*, v. 16 n. 12-13, pp. 941-946.

Giardina, C. R., and Dougherty, E. R. (1988). *Morphological Methods in Image and Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall.

Gonzales, R. C. & Woods, R. E. (1992). *Digital Image Processing*. Reading, MA: Addison-Wesley.

Goodacre, R. (1994). Characterization and quantification of microbial systems using pyrolysis mass spectrometry: Introducing neural networks to analytical pyrolysis. *Microbiology Europe*, 2(2):16-22.

Goodhill, G. J. & Sejnowsky, T. J. (1997). A unifying objective function for topographic mappings. *Neural Computation*, vol. 9, no. 6, pp. 1291-1303.

Gordon, A. D. & Henderson, J. (1977). An algorithm for Euclidean sum of squares classification. *Biometrics*, vol. 33, pp. 355-362.

Gordon, A. D. (1981). *Classification: Methods for the Exploratory Analysis of Multivariate Data*, New York, Chapman-Hall.

Gordon, A. D. (1987). A review of hierarchical classification. *J. Royal Statistical Society*, A, vol. 150, Part 2, pp. 119-137.

Gordon, A. D. (1996). Hierarchical classification. In: Arabie, P., Hubert, L. J., De Soete, G. (Eds). *Clustering and Classification*, pp. 65-111. World Scientific Pub Co: Singapore.

Gower, J. C. (1967). A comparison of some methods of cluster analysis. *Biometrics*, vol. 23, pp. 623-637.

Grossberg, S. (1976). Adaptive pattern classification and universal recording: I. Parallel development and coding of neural detectors. *Biological Cybernetics*, vol. 23, pp. 121-134.

Grossberg, S. (1995). Are there universal principles of brain computation? In: Mira, J. & Sandoval, F. (Eds.), *Proc. of the Intl. Workshop on Artificial Neural Networks*, pp. 1-6, Spain, Springer.

Guha, S., Rastogi, R., Shim, K. (1998). CURE: An efficient clustering algorithm for large databases. *Proc. of the ACM SIGMOD International Conference on Management of Data*, pp. 73-84, Seattle, WA.

Hagiwara, M. (1996). Self-organizing feature map with a momentum term. *Neurocomputing*, 10(1):71-81.

Hamad, D., Firmin, C. & Postaire, J. (1996). Unsupervised pattern classification by neural networks. *Mathematics and Computers in Simulation*, v. 41, pp. 109-116.

Hansen, P. and Jaumard, B. (1997). Cluster analysis and mathematical programming. *Mathematical Programming*, vol. 79, pp. 191-215.

Haralick, R. M., Sternberg, S. R., and Zhuang, X. (1987). Image analysis using mathematical morphology. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 532-550.

Hartigan, J.A. (1975). *Clustering Algorithms*, New York: John Wiley & Sons.

Hartigan, J.A. (1977), Distribution Problems in Clustering. In: *Classification and Clustering*, ed. J. Van Ryzin, New York: Academic Press, Inc.

Hartigan, J.A. (1978), Asymptotic Distributions for Clustering Criteria, *Annals of Statistics*, 6, 117-131.

Hartigan, J.A. (1981), Consistency of Single Linkage for High-Density Clusters, *Journal of the American Statistical Association*, 76, pp. 388-394.

Hartigan, J.A. (1985), Statistical Theory in Clustering, *Journal of Classification*, 2, 63-76.

Hausknecht, M. (1988). Basistypen flexibler Automation. *VDI-Z*, v. 130, n. 12 , pp. 30-35

Haykin, S. (1999). Neural Networks, a Comprehensive Foundation. 2nd edition, Prentice-Hall.

Hebb, D. O. (1949). *The Organization of Behavior: A Neuropsychological Theory*. New York: Wiley.

Hecht-Nielsen, R. (1987). Counterprogagation networks. *Applied Optics*, vol. 26, no. 23, pp. 4979-4984.

Heskes, T. & Kappen, B. (1993). Error potential for self-organization. In *Proc. ICNN'93, Int. Conf. on Neural Networks*, volume III, pages 1219-1223, Piscataway, NJ.

Hesselroth, T., Sarkar, K., van der Smagt, P., & Schulten, K. (1993). Neural network control of a pneumatic robot arm. *IEEE Trans. on Syst. , Man and Cyb.*, 24:28-37, 1993.

Himes, G.S.; Iñigo, R.M. (1992). Automatic Target Recognition Using a Neocognitron. *IEEE Trans. Knowledge and Data Engineering*, v. 4, n. 2.

Hopfield, J. J.; D.W. Tank (1986). Computing with neural circuits: a model. *Science magazine*, v. 233, p. 625-633.

Hubel, D. and Wiesel, T. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106 - 154.

Ienne, P. & Viredaz, M. A. (1994). Implementation of Kohonen's self-organising maps on MANTRA I. *In Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, pp. 273-9, Los Alamitos, CA, USA.

Jain, A.K. & Dubes, R.C. (1988). *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, N.J.

Jambu, M. (1978). *Classification automatique pour l'analyse des donnés (Tome 1).* Paris: Dunod.

Jambu, M. & Lebeaux, M. O. (1983). *Cluster Analysis and Data Analysis*; Elsevier Science: Amsterdan.

Jantsch, E. (1980). *The Self-Organizing Universe.* Pergamon Press, Oxford, New York.

Jobson, J. D. (1991). *Applied Multivariate Data Analysis : Regression and Experimental Design,* vol. 1. New York: Springer Verlag.

Jobson, J. D. (1992). *Applied Multivariate Data Analysis : Categorical and Multivariate Methods*, vol. 2. New York: Springer Verlag.

Johnson, R. A. & Wichern, D. W. (1998). *Applied Multivariate Statistical Analysis.* 4th Ed., Englewood Cliffs, NJ: Prentice-Hall.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, vol. 32, pp. 241-254.

Jollife, I. T. (1986). Principal Component Analysis. New York: Springer.

Jones, M. & Vernon, D. (1994). Using neural networks to learn hand-eye co-ordination. *Neural Computing & Applications*, 2(1):2-12.

Kandel, A. (1982). *Fuzzy techniques in pattern recognition.* New York: John Wiley.

Kandel, A. (1986). *Fuzzy mathematical techniques with applications.* Reading, MA: Addison-Wesley.

Kangas, J.A., Kohonen, T. and Laaksonen, J.T. (1990). Variants of Self-Organizing Maps. *IEEE Trans. on Neural Networks*, vol., no. 1, pp. 93-99.

Kangas, J. & Kohonen, T. (1996). Developments and applications of the self-organizing map and related algorithms. *Mathematics and Computers in Simulation*, 41(1-2):3-12.

Kaski, S., Lagus, K., Honkela, T., & Kohonen, T. (1998). Statistical aspects of the WEBSOM system in organizing document collections. *Computing Science and Statistics*, 29:281-290.

Kaski, S. & Kohonen, T. (1996). Exploratory data analysis by the self-organizing map: Structures of welfare and poverty in the world. In Refenes, A.-P., Abu-Mostafa, Y., Moody, J. & Weigend, A. (eds.). *Neural Networks in Financial Engineering*, pp. 498-507.World Scientific, Singapore

Kaski, S. and Lagus, K. (1996) Comparing self-organizing maps. In: von der Malsburg, C., von Seelen, W., Vorbrüggen, J. C., & Sendhoff, B. (Eds.), *Proceedings of ICANN'96, International Conference on Artificial Neural Networks, Lecture Notes in Computer Science*, vol. 1112, pages 809-814. Springer, Berlin.

Kaski, S., Kangas, J. & Kohonen, T. (1998), Bibliography of Self-Organizing Map (SOM) Papers: 1981--1997, *Neural Computing Surveys*, 1: 102-350. Disponível eletrônicamente em http://www.icsi.berkeley.edu/~jagota/NCS/.

Kaski, S., Nikkilä, J. & Kohonen, T. (1998). Methods for interpreting a self-organized map in data analysis. In: *Proc. of ESANN'98, 6th Eur. Symp. on Artif. Neural Networks*. Brussels, Belgium: D-Facto, pp. 185-190.

Kaufman, L. & Rousseeuw, P. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley: New York.

Kirk, J. & Zurada, J. (1999). Algorithms for improved topology preservation in self-organizing maps. In: *Proc. IEEE Conf. on Systems, Man & Cybernetics*, vol. III, pp. 396-400. Tokyo, Japan: IEEE.

Kiviluoto, K. (1996). Topology preservation in Self-Organizing Maps. In: *Proc. ICNN'96 - Intl. Conf. on Neural Networks*, vol. 1, pp. 249-254. Piscataway, NJ: IEEE.

Klir, G. J. & Yuan, B. (1995). *Fuzzy sets and fuzzy logic: Theory and applications*. Upper Saddle River, NJ: Prentice-Hall.

Kohonen, T. (1982a). Self-organizing formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59-69.

Kohonen, T. (1982b). Analysis of a simple self-organizing process. *Biological Cybernetics*, 44(2):135-140.

Kohonen, T. (1984). *Self-Organization and Associative Memory*. Springer, Berlin, Heidelberg, 1984. (3rd ed. 1989).

Kohonen, T. (1988). The 'neural' phonetic typewriter. *Computer*, 21(3):11-22.

Kohonen T. (1989a). *Self-Organization and Associative Memory*, 3rd edition, Springer Verlag, Berlin, Germany

Kohonen (1989b). Speech recognition based on topology-preserving neural maps. In Igor Aleksander, editor, *Neural Computing Architectures*, pages 26-40. North Oxford Academic Publishers/Kogan Page, Oxford, UK.

Kohonen, T. (1990). The self-organizing map. *Proc. IEEE*, 78:1464-1480, 1990.

Kohonen, T. (1992). Learning-Vector Quantization and the Self-Organizing Map. In J. G. Taylor and C. L. T. Mannion, editors, Theory and Applications of Neural Networks, pages 235-242, London, UK, Springer.

Kohonen, T. (1993). Physiolocigal interpretation of the self-organizing map algorithm. *Neural Networks*, 6(7):895-905.

Kohonen, T. (1994). Physiological model for the Self-Organizing Map. *In Proc. WCNN'94, World Congress on Neural Networks*, volume III, pages 97-102, Hillsdale, NJ, Lawrence Erlbaum.

Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J. & Torkkola, K. (1996a). *LVQ_PAK: The Learning Vector Quantization program package*. Report A30, Helsinki University of Technology, Laboratory of Computer and Information Science, January.

Kohonen, T., Hynninen, J., Kangas, J. & Laaksonen, J. (1996b). *SOM_PAK: The Self-Organizing Map program package*. Report A31, Helsinki University of Technology, Laboratory of Computer and Information Science, January.

Kohonen, T., Oja, E., Simula, O., Visa, A. & Kangas, J. (1996c). Engineering applications of the self-organizing map. *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1358-84.

Kohonen, T. (1997a). *Self-Organizing Maps*, 2nd Ed., Springer-Verlag: Berlin.

Kohonen, T. (1997b). Exploration of very large databases by self-organizing maps. In: *Proc. of the 1997 IEEE Intl. Conf. on Neural Networks*, pp. PL1-6, Houston, Texas.

Kohonen, T (1998). Self-Organization of Very Large Document Collections: State of the Art, in Niklasson L., et al.(eds.), *Proceedings of the 8th International Conference on Artificial Neural Networks*, Skoevde, Sweden, Springer, Berlin, 2 vols., pp.65-74.

Koikkalainen, P. (1994). Progress with the tree-structured self-organizing map. In A. G. Cohn, editor, *Proc. ECAI'94, 11th European Conf. on Artificial Intelligence*, pp. 211-215, New York, John Wiley & Sons.

König, A., Geng, X. & Glesner, M. (1993). Hardware implementation of Kohonen's feature map by scalar and SIMD-array processors. In Stan Gielen and Bert Kappen, editors, *Proc. ICANN'93, Int. Conf. on Artificial Neural Networks*, pages 1046-1049, London, UK.

Konstatinides, K. & Rasure, J. (1994). The Khoros Software Development Environment for Image and Signal Processing. *IEEE Trans. on Image Processing*, vol. 3, no. 3, pp. 243-252.

Kopp, B. (1978). Hierarchical classification II: Complete linkage method. *Biometrical Journal*, vol. 20, pp. 597-602.

Krishapuram, R. & Keller, J. M. (1994). Fuzzy Set Theoretic Approach to Computer Vision: An Overview. In: *Fuzzy Tecnology and Applications*, Marks II, R.J., (ed.). New Jersey, IEEE Press.

Kröse, B. J. A. & Eecen, M. (1994). Self-learning maps for path planning in sensor space. In Maria Marinaro and Pietro G. Morasso, editors, *Proc. ICANN'94, Int. Conf. on Artificial Neural Networks*, volume II, pages 1303-1306, London, UK, 1994. Springer.

Lampinen, J. & Oja, E. (1992). Clustering properties of hierarchical self-organizing maps. *J. Mathematical Imaging and Vision*, 2(2-3):261-272.

Lance, G.N. & Williams, W. T. (1967). A general theory of classificatory sorting strategies: 1. Hierarchical systems. *Computer Journal*, vol. 9, pp. 373-380.

Lancini, R. (1994). Image vector quantization by neural networks. In Ben Yuhas and Nirwan Ansari, editors, *Neural Networks in Telecommunications*, pages 287-303, Dordrecht, Netherlands. Kluwer Academic Publishers.

Lee, T. & Peterson, A. M. (1990). Adaptive vector quantization using a self development neural network. *IEEE Journal on Selected Areas in Communications*, v. 8, pp. 1458-1471.

Li, T., Tang, Y., Suen, S., & Fang, L. (1992). A structurally adaptive neural tree for recognition of a large character set. In: *Proc. of the 11th IAPR Intl. Joint Conf. on Pattern Recognition*, vol. II, pp. 187-190.

Li, T., Fang, L., & Li, Q.-Q. (1993). Hierarchical classification and vector quantisation with neural trees. *Neurocomputing*, 5, 119-139.

Linde, Y., Buzo, A., and Gray, R. M. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communication*, vol. COM-28, pp. 84-95.

Lorr, M. (1983). *Cluster Analysis for Social Scientists*. Jossey-Bass Pub; ISBN: 0875895662.

Lu, C. C. & Shin, Y. H. (1992). A neural network based image compression system. *IEEE Trans. on Consumer Electronics*, 38(1):25-29.

Lund, K.; Christiansen, E.H.; Lund, B.; Pedersen, A.K. (1998). Recovery of beat-to-beat variations of QRS. *Medical & Biological Engineering & Computing*, v. 36, n. 4, pp. 438-444

MacQueen, J.B. (1967), Some Methods for Classification and Analysis of Multivariate Observations, In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1, pp. 281-297.

Mandel, I. D. e Chernyl, L. B. (1988). Experimental comparison of cluster analysis of algorithms. Automation and Remote Control, v. 49, pp. 87-94.

Mao, J. & Jain, A.K. (1996). A Self-Organizing Network for Hyper-Ellipsoidal Clustering (HEC). *IEEE Trans. Neural Networks*, vol. 7, no. 1, pp. 16-29.

Mangiameli, P., Chen, S.K., & West, D. (1996). A comparison of SOM neural network and hierarchical clustering methods. *European J. Operational Research*, v. 93, 402-417.

Mardia, K. V., Kent, J. T., & Bibby, J.M. (1979). *Multivariate Analysis*. San Diego, CA: Academic Press.

Maren, A.J., Harston, C.J., & Pap, R.M. (1990). *Handbook of neural computing applications*. San Diego. Academic Press.

Marr, D. (1982). *Vision*. New York: W. H. Freeman.

Marriott, F.H.C. (1982). Optimization methods of cluster analysis. *Biometrika*, vol. 69, pp. 417-421.

Mascarenhas, N. D. A. & Velasco, F. R. D. (1989). *Processamento digital de imagens*. 2a Ed., I Escola Brasileiro-Argentina de Informática. Buenos Aires: Ed. Kapelusz.

Massart, D.L. and Kaufman, L. (1983), The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis, New York: John Wiley & Sons, Inc.

Masters, T. (1995) Advanced Algorithms for Neural Networks: A C++ Sourcebook. John Wiley, New York.

Matheron, G. (1975). *Random Sets and Integral Geometry*. Wiley: New York

McLachlan, G.O., and Basford, K.E. (1988). *Mixture Models*. New York: Marcel Dekker.

Meijster, A. & Roerdink, J. B. (1996). Computation of watersheds based on parallel graph algorithms. In: Maragos, P., Schafer, R. & Butt, M. (Eds). *Mathematical Morphology and its Applications to Image and Signal Processing,* pp. 305-312. Boston, MA: Kluwer Academic Press

Merelo, J. J., Andrare, M. A., Prieto, A., & Morán, F. (1994). Proteinotopic feature maps. *Neurocomputing*, 6(1):443-454.

Merkl, D. (1997). Exploration of Document Collections with Self-Organizing Maps: A Novel Approach to Similarity Representation. In: *Proc. of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery* (PKDD'97), pp. 101-111. Trondheim. Norway. Springer-Verlag

Merkl, D. (1998). Text classification with self-organizing maps: Some lessons learned, *Neurocomputing*, vol. 21, pp. 61-77.

Merkl, D. & Rauber, A. (1997a). Cluster Connections: A visualization technique to reveal cluster boundaries in self-organizing maps. In: *Proc 9th Italien Workshop on Neural Nets* (WIRN97), Vietri sul Mare, Italy: Springer-Verlag.

Merkl, D. & Rauber, A. (1997b). Alternative Ways for Cluster Visualization in Self-Organizing Maps. In: *Proc of the Workshop on Self-Organizing Maps* (WSOM97), Helsinki, Finland.

Meyer, F., and Beucher, S. (1990). Morphological Segmentation, *Journal of Visual Comm. & Image Representation*, vol. 1, pp. 21-46.

Meyer, F. (1993). Gradients and watershed lines. In: Serra, J. & Salembier, P. (Eds.), *Proc. Workshop on Mathematical Morphology and its Applications to Signal Processing*, Barcelona, pp. 70-75.

Michaud, P. (1997). Clustering techniques. *Future Generation Computer Systems*, vol. 13, pp. 135-147.

Miikulainen R. (1990). Script Recognition with Hierarchical Feature Maps, *Connection Science*, vol. 2, pp. 83-101

Miikulainen R. (1991). Self-Organizing Process Based on Lateral Inhibition and Synaptic Resource Redistribution, *Proceedings of the International Conference on Neural Networks* (ICANN'91), pp.415-420, Espoo, Finland

Miikulainen, R. (1993a). Integrated Connectionist Models: Building AI Systems on Subsymbolic Foundations. In: Honvar, V. & Uhr, L. (eds.). *Integrating Symbol Processors and Connectionist Networks for Artificial Intelligence and Cognitive Modeling*, New York: Academic Press.

Miikkulainen, R. (1993b). *Subsymbolic Natural Language Processing: An Integrated Model of Scripts, Lexicon, and Memory.* MIT Press, Cambridge, MA, 1993.

Milligan, G.W. (1980), An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms, *Psychometrika*, 45, 325-342.

Milligan, G.W. and Cooper, M.C. (1985), An Examination of Procedures for Determining the Number of Clusters in a Data Set, *Psychometrika*, 50, 159-179.

Minnix, J.I., Mcvey, E.S. & Iñigo, R.M. (1992). A multilayered self-organizing artificial neural network for invariant pattern recognition. *IEEE Trans. on Knowledge and Data Engineering*, v. 4, n. 2, p. 162-167.

Missaoui, R. e Frasson, C. (1989). Utilization of database analysis techniques for optimization of relational queries. *Information Systems and Operational Research*, v. 27 n. 3, pp. 338-359.

Mitchison, G. (1995). A type of duality between Self-Organizing Maps and minimal wiring. *Neural Computation*, 7(1):25-35.

Morasso, P. & Sanguineti, V. (1994). Cortical representation of external space. In Maria Marinaro and Pietro G. Morasso, editors, Proc. ICANN'94, *Int. Conf. on Artificial Neural Networks*, volume II, pages 1247-1252, London, UK, Springer.

Morrison, D. G. (1967). Measurement problems in cluster analysis. *Management Science*, vol. 13, pp. 775-780.

Murtagh, F. (1983). A Survey of Recent Advances in Hierarchical Clustering Algorithms. *The Computer Journal*, vol. 26, n. 4, pp. 354-359

Murtagh, F. (1995). Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering. *Pattern Recognition Letters*, v. 16, pp. 399-408

Murtagh, F. (1996). The Kohonen Self-Organizing Map Method: An Assessment. *Jornal of Classification*, vol. 12, pp. 165-190.

Najman, L. & Schmitt, M. (1996). Geodesic Saliency of Watershed Contours and Hierarchical Segmentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 18, 1163-1173.

Nevo, I., Guez, A., Ahmed, F., e Roth, J. V. (1991). System theoretic approach to medical diagnosis. *Proceedings of the 4th IEEE Annual Symposium on Computer-Based Medical Systems*, p 94-96, Piscataway, NJ.

Nicolis, G. & Prigogine, I. (1989). *Exploring Complexity*. W. H. Freeman: New York.

Noguet, D., Merle, A. & Lattard, D. (1996). A data dependent architecture based on seeded region growing strategy for advanced morphological operations. In: Maragos, P., Schafer, R. & Butt, M. (Eds). *Mathematical Morphology and its Applications to Image and Signal Processing*, pp. 235-243. Boston, MA: Kluwer Academic Press

Obermayer, K., Ritter, H., & Schulten, K. (1991). Development and spatial structure of cortical feature maps: A model study. In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing Systems* 3, pages 11-17. Morgan Kaufmann, San Mateo, CA.

Obermayer, K., Schulten, K., & Blasdel, G. G. (1992). A comparison of a neural network model for the formation of brain maps with experimental data. In John E. Moody, Stephen J. Hanson, and Richard P. Lippmann, editors, *Advances in Neural Information Processing Systems*, Morgan Kaufmann, San Mateo, CA.

Obermayer, K. & Blasdel, G. G. (1997). Singularities in primate orientation maps. *Neural Computing*, 9:555-576.

Oja, E. (1992). Self-organizing maps and computer vision. In Harry Wechsler, editor, *Neural Networks for Perception*, vol. 1: *Human and Machine Perception*, pages 368-385. Academic Press, New York, NY.

Oja, E. (1995). *Neural Networks for Chemical Engineers*, volume 6 of *Computer-Aided Chemical Engineering*, chapter 2, Unsupervised neural learning. Amsterdam: Elsevier.

Pal, N.K., Bezdek, J. & Tsao, E. C. (1992). Improving convergence and performance of Kohonen's self-organizing sceme. In SPIE Vol. 1710, *Science of Artificial Neural Networks*, pages 500-509, Bellingham, WA, SPIE.

Pal, N., Bezdek, J. & Tsao, E. C. (1993). Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Trans. on Neural Networks*, 4(4):549-557.

Pal, N.K. & Bezdek, J.C. (1995). On cluster validity for the fuzzy c-means model. *IEEE Trans. on Fuzzy Systems*, 3 (3), 370-379.

Pal, N.R. & Pal, S. K. (1993). A review on image segmentation techniques. *Pattern Recognition*, v. 26, n. 9, p. 1277-1294.

Pal, S. (1994). Fuzzy sets in image processing and recognition. In: *Fuzzy Tecnology and Applications,* Marks II, R.J., (ed.). New Jersey: IEEE Press.

Pan, H.-L., & Chen, Y.-C. (1992). Liver tissues classification by artificial neural networks. *Pattern Recognition Letters*, 13(5):355-368.

Park, Y. (1994). A comparison of neural classifiers and linear tree classifiers: Their similarities and differences. *Pattern Recognition*, vol. 27, no. 11, pp. 1493-1503.

Parker, J. R. (1994). *Practical Computer Vision Using C.* New York: Wiley.

Parker, J.R. (1997). *Algorithms for image processing and computer vision.* Wiley: New York.

Pedrycz, W. (1990). Fuzzy sets in pattern recognition: methodology and methods. *Pattern Recognition*, vol. 23, n. 1/2, p. 121-146.

Pessoa Jr., O. (1996). Medidas sistêmicas e organização. In: Debrun et al. (1996) Eds. *Auto-Organização: Estudos Interdisciplinares*. Pp. 129-161. CLE, Unicamp. Campinas, SP.

Pinkowski, B. (1989). CLUSTERT - a simulation-based expert system. *Simulation*, v. 52, pp. 179-185

Pomierski, T., Gross, H. M. & Wendt, D. (1993). A distributed multicolumnar system for primary cortical analysis of real-world scenes. In Stan Gielen and Bert Kappen, editors, *Proc. ICANN'93, Int. Conf. on Artificial Neural Networks*, pp. 142-147, London, UK, Springer.

Popchev, I. e Peneva, V. (1988). Cluster - a package for cluster analysis. In: *Proc. Intl. Conference of the IEEE Engineering in Medicine and Biology Society*, p 1466-1467, Piscataway, NJ.

Racz, J., & Klotz, T. (1991). Knowledge representation by dynamic competitive learning techniques. *Proc. of the SPIE*, vol. 1469, 778-783.

Ramussen, E. M. (1990). Clustering algorithms in information retrieval. In: Frakes, W., e Beaza-Yates, R. (Eds.). *Information Retrieval: Data structures and Algorithms*. Prentice-Hall, Englewood Cliffs, NJ.

Ramussen, E. M. e Willett, P. (1989). Efficiency of hierarchic agglomerative clustering using the ICL Distributed Array Processor. *Journal of Documentation*, v. 45m n. 1, pp. 1-24

Rauber, A. & Merkl, D. (1999). Automatic Labeling of Self-Organizing Maps: Making a Treasure-Map Reveal its Secrets. In: *Proceedings of the 3. Pacific-Asia Conference on Knowledge Discovery and Data Mining* (PAKDD'99), Bejing, China, April 26-28. Lecture Notes in Artificial Intelligence, Springer Verlag.

Redner, R. A. & Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, vol. 26, pp. 195-239.

Richard, R.D. & Lippman, R.P. (1991). Neural network classifiers estimate Bayesian a posteriori probabilities. *Neural Computation*, v. 1, p. 281-294.

Ripley, B.D. (1994). Neural Networks and Related Methods for Classification. *Journal of the Royal Statistical Society B*, v. 56, n. 3, p. 409-456.

Ripley, B.D. (1996). *Pattern Recognition and Neural Networks*, Cambridge University Press.

Rissanen, J. (1978). Modeling by shortest data description, *Automatica*, v. 14, p. 465-471.

Rissanen, J. (1989). *Stochastic complexity in statistical inquiry*. World Scientific, Teaneck, New Jersey.

Ritter, H. J. (1989). Combining self-organizing maps. *In Proc. IJCNN'89, Int. Joint Conf. on Neural Networks*, Washington DC, volume II, pages 499-502, Piscataway, NJ.

Ritter, H. & Kohonen, T. (1989a). *Self-organizing semantic maps.* Report, Helsinki Univ. of Technology, Lab. of Computer and Information Science, Espoo, Finland.

Ritter, H. & Kohonen, T. (1989b). Self-organizing semantic maps. *Biological Cybernetics*, 61(4):241-254, 1989.

Ritter, H., Martinetz, T., & Schulten, K. (1992). Neural Computation and Self-Organizing Maps: An Introduction. Addison-Wesley, Reading, MA.

Ritter, H. & Schulten, K (1986). On the stationary state of Kohonen's self-organizing sensory mapping. *Biological Cybernetics*, 54:99-106, 1986.

Ritter, H., & Schulten, K. (1988). Kohonen self-organizing maps: exploring their computational capabilities. In *Proc. ICNN'88 Int. Conf. on Neural Networks*, volume I, pages 109-116, Piscataway, NJ.

Robbins, H. & Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, vol. 22, pp. 400-407.

Rumelhart, D. E., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, vol. 9, pp. 75-112.

Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. (1986). Learning internal representations by error propagation. In: D.E. Rumelhart e J.L. McClelland, Eds., *Parallel distributed processing: explorations in the microstructure of cognition*, v. 1, Cambridge, MA. MIT Press.

Rumelhart, D., McCllelland, J., and The PDP Research Group (1986). Feature discovery by competitive learning. In: Rumelhart et al. (1986). *Parallel Distributed Processing*. Cambridge, MA. MIT Press.

Russ, J. C. (1995). *The Image Processing Handbook*, 2nd Ed., Boca Raton, FL: CRC Press.

Sammon, J. W. (1969). A Non-Linear Mapping for Data Structure Analysis. *IEEE Trans. in Computers*, vol. 18, pp. 401-409.

SAS Institute (1996). *Manual do usuário do sistema SAS*. SAS Institute Inc, Cary, NC, USA.

Saxon, J. B. (1991). *Simulating sensorimotor systems with cortical topology*. Master's thesis, Texas A&M University, Computer Science Department, College Station, Texas.

Scholtes, J. C. (1991). Unsupervised learning and the information retrieval problem. In *Proc. IJCNN'91, Int. Joint Conf. on Neural Networks*, volume I, pages 95-100, Piscataway, NJ.

Scoltock, J. (1982). A Survey of the Literature of Cluster Analysis. *The Computer Journal*, vol. 25 n.1, p. 130-134.

Scott, A.J. and Symons, M.J. (1971), Clustering Methods Based on Likelihood Ratio Criteria, *Biometrics*, 27, 387-397.

Serra, J. (1982). *Image Analysis and Mathematical Morphology*. Academic Press: London.

Serrano, C., Martín, B. & Gallizo, J. L. (1993). Artificial neural networks in financial statement analysis: Ratios versus accounting data. *In Proc. 16th Annual Congress of the European Accounting Associatian.*

Shepard, N. R. (1962a). The analysis of proximity: multidimensional scaling with an unknown distance function I. *Psychometrika*, vol. 27, pp. 125-139.

Shepard, N. R. (1962b). The analysis of proximity: multidimensional scaling with an unknown distance function II. *Psychometrika*, vol. 27, pp. 219-246.

Siemon, H. P. & Ultsch, A. (1990). Kohonen networks on transputers: implementation and animation. In *Proc. INNC-90 Int. Neural Network Conf.*, pp. 643-646, Dordrecht, Netherlands, 1990. Kluwer.

Sirosh J., & Miikkulainen R. (1993). How Lateral Interaction Develops in a Self-Organizing Feature Map, *Proceedings of the IEEE International Conference on Neural Networks* (ICNN'93), San Francisco, CA, USA

Sneath, P.H.A. (1957). The application of computers to taxonomy. *J. Gen. Microbiol.*, 17, pp. 201-226.

Sokal, R.R. and Sneath, P.H.A. (1973), *Numerical Taxonomy*, San Francisco: W.H. Freeman.

Späth, H. (1980). *Cluster Analysis Algorithms: for data reduction and classification of objects*. Ellis Horwood: Chichester, West Sussex, England.

Späth, H. (1985). *Cluster Dissection and Analysis*. Ellis Horwood: Chichester, England.

Speckmann, H., Raddatz, G. & Rosenstiel, W. (1994). Considerations of geometrical and fractal dimension of SOM to get better learning results. In: Marinaro, M. & Morasso, P. (Eds.), *Proc. ICANN'94, Int. Conf. on Artificial Neural Networks*, volume I, pp. 342-345.

Su, M.-C., DeClaris, N. & Liu, T.-K. (1997). Application of neural networks in cluster analysis. In: *Proc. of the 1997 IEEE Intl. Conf. on Systems, Man, and Cybernetics*, pp. 1-6.

Sutton III, G. G., Reggia, J. A., Armentrout, S. L., & D'Autrechy, C.L. (1994). Cortical map reorganization as a competitive process. *Neural Computation*, 6(1):1-13.

Symons, M.J. (1981), Clustering Criteria and Multivariate Normal Mixtures, *Biometrics*, 37, pp. 35-43.

Tenenbaum, A. M., Langsam, Y. & Augenstein, M. J. (1995). *Estruturas de dados usando C*. São Paulo: Makron Books.

Thode, H.C.Jr., Mendell, N.R., and Finch, S.J. (1988), Simulated percentage points for the null distribution of the likelihood ratio test for a mixture of two normals, *Biometrics*, vol. 44, pp. 1195-1201.

Timmins, J., Neal, M. & Hunt, J. (1999). Data Analysis with Artificial Immune Systems, Cluster Analysis and Kohonen Networks : Some Comparisons. In: *Proc. of the IEEE International Conference on Systems, Man and Cybernetics*, Tokyo, Japan.

Ting, C. & Chuang, K-C. (1993). Adaptive algorithm for neocognitron to recognize analog images. *Neural Networks*, v. 6, n. 2, p. 285-299

Titterington, D.M., Smith, A.F.M., and Makov, U.E. (1985*), Statistical Analysis of Finite Mixture Distributions*, New York: John Wiley & Sons, Inc.

Tokutaka, H., Yoshihara, K., Fujimura, K., Obu-Cann, K. & Iwamoto, K. (1999). Application of self-organizing maps to chemical analysis. Applied Surface Science, vol. 145, pp. 59-63.

Torgerson, W. S. (1952). Multidimensional Scaling I. Theory and Method. *Psychometrika*, vol. 17, pp. 401-419.

Tolat, V. V. (1990). An analysis of Kohonen's self-organizing maps using a system of energy functions. *Biological Cybernetics*, 64(2):155-164.

Tryon, R. C. & Bailey, D. E. (1970). *Cluster Analysis*. McGraw-Hill, New York.

Turner, M., Austin, J., Allinson, N., & Thompson, P. (1993). Chromosome location and feature extraction using neural networks. *Image and Vision Computing*, 11(4):235-239.

Ultsch, A. (1993a). Knowledge Extraction from Self-Organizing Neural Networks. In: O. Opitz et al. (Eds). *Information and Classification*. Springer, Berlin, 301-306.

Ultsch, A. (1993b). Self-Organizing Neural Networks for Visualization and Classification. In: O. Opitz et al. (Eds). *Information and Classification*. Springer, Berlin, 307-313. 1993.

Ultsch, A. (1995). Self-Organizing Neural Networks perform different from statistical k-means clustering. *Gesellschaft für Klassifikation*, Basel.

van Gils, M. J. & Cluitsman, P. J. M. (1993). Assessing the latence of peak pa in auditory evoked potential using neural networks. In Stan Gielen and Bert Kappen, editors, *Proc. ICANN'93, Int. Conf. on Artificial Neural Networks*, page 1015, London, UK.

Villmann, T., Der, R., Martinetz, T. (1994). A new quantitative measure of topology preservation in Kohonen's feature maps. In: *Proc. ICNN'94, IEEE Int. Conf. on Neural Networks*, pp. 645-648. IEEE Service Center, Piscataway, NJ.

Villmann, T., Der, R., Herrmann, M. & Martinetz, T. (1997) Topology Preservation in Self-Organizing Feature Maps: Exact Definition and Measurement. *IEEE Trans. on Neural Networks*, vol. 8, no. 2, pp. 256-266.

Visa, A. (1990a). *Texture Classification and Segmentation Based on Neural Network Methods.* PhD thesis, Helsinki University of Technology, Espoo, Finland, 1990.

Visa, A. (1990b). Identification of stochastic textures with multiresolution features and self-organizing maps. In *Proc. 10ICPR, Int. Conf. on Pattern Recognition*, pages 518-522, Piscataway, NJ, 1990.

Visa, A. (1992). Industrial applications of artificial neural networks in Finland. *In Proc. DECUS Finland ry. Spring Meeting*, pages 323-332, Helsinki, Finland.

Visa, A. (1994). Texture segmentation based on neural networks. *In Proc. 3rd Int. Conf. on Fuzzy Logic, Neural Nets and Soft Computing*, pages 145-148, Iizuka, Japan, 1994. Fuzzy Logic Systems Institute.

von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85 - 100.

Waller, N.G., Kaiser, H.A., Illian, J.B. & Manry M. (1998). A comparison of the classification capabilities of the 1-dimensional Kohonen neural network with two partitioning and three hierarchical cluster analysis algorithms, *Psychometrika*, vol. 63, pp. 5-22.

Walter, J. & Schulten, K. I (1993). Implementation of self-organizing neural networks for visuo-motor control of an industrial robot. *IEEE Transactions on Neural Networks*, 4(1):86-96.

Wan, W. & Fraser D. (1993). *A Self-Organizing Neural Network for Contextual Analysis of Spatial Patterns of Multisource Data*, Proceedings of the Conference on Digital Image Computing, Techniques and Applications (DICTA'93), Sydney, Australia

Wan, W., Fraser, D. (1994a). *Multiple Kohonen Self-Organizing Maps: Supervised and Unsupervised Formation with Application to Remotely Sensed Image Analysis*, Proceedings of the 7th Australian Remote Sensing Conference (7ARSC), Melbourne, Australia

Wan, W., Fraser, D. (1994b). *A Self-Organizing Neural Network Framework for Unsupervised and Supervised Classification*, Proceedings of the 7th Australian Remote Sensing Conference (7ARSC), Melbourne, Australia

Wan, W., Fraser, D. (1994c). *A Self-Organizing Neural Network Framework for Multisource Data and Contextual Analysis*, Proceedings of the 7th Australian Remote Sensing Conference (7ARSC), Melbourne, Australia

Ward, J.H. (1963), Hierarchical Grouping to Optimize an Objective Function, *Journal of the American Statistical Association*, vol. 58, pp. 236-244.

Weinstein, J. N., Myers, T. G., Kan, Y., Paull, K. D., Zaharevitz, D. W., & van Osdol, and K. W. (1995). An 'information-intensive' strategy for drug discovery at the national cancer institute: The role of neural networks. *In Proc. WCNN'95*, pp. 750-753

Whittington, G. & Spracklen, C. T. (1994). An efficient multiprocessor mapping algorithm for the Kohonen feature map and its derivative models. *In Proc. ICNN'94, Int. Conf. on Neural Networks*, pp. 17-21, Piscataway, NJ.

Willshaw, D.J. & von der Malsburg, C. (1976). How patterned neural connections can be set up by self-organization. *Proc. of the Royal Society of London, Series B*, vol. 194, pp. 431-445.

Wilson, C. L. (1994). Self-organizing neural network system for trading common stocks. In *Proc. ICNN'94, Int. Conf. on Neural Networks*, pp. 3651-3654, Piscataway, NJ.

Wiskott, L. & von der Malsburg, C. (1996). Face Recognition by Dynamic Link Matching. In Sirosh, J., Miikkulainen, R., and Choe, Y., editors, *Lateral Interactions in the Cortex: Structure and Function*. The UTCS Neural Networks Research Group, Austin, TX. Electronic book, ISBN 0-9647060-0-8, http://www.cs.utexas.edu/ users/nn/web-pubs/htmlbook96, Chapter 11.

Wolfe, J.H. (1970), Pattern Clustering by Multivariate Mixture Analysis, *Multivariate Behavioral Research*, 5, 329-350.

Wolfe, J.H. (1978), Comparative Cluster Analysis of Patterns of Vocational Interest, *Multivariate Behavioral Research*, 13, 33-44.

Wong, M.A. and Lane, T. (1983), "A k^{th} Nearest Neighbor Clustering Procedure, *Journal of the Royal Statistical Society*, Series B, vol. 45, pp. 362-368.

Wyler, K. (1993). Self-organizing process mapping in a multiprocessor system. *In Proc. WCNN'93, World Congress on Neural Networks*, volume II, pages 562-566, Hillsdale, NJ, 1993. Lawrence Erlbaum.

Yin, H. & Allinson, N. M. (1993). On the distribution of feature space in self-organisng mapping and convergence accelerating by a Kalman filter. In J. Mira, J. Cabestany, and A Prieto, editors, *New Trends in Neural Computation*, pages 291-96, Berlin, Heidelberg, Springer.

Yovits, M. & Cameron, S. (eds.) (1960). *Self-Organizing Systems*. Pergamon Press, Oxford.

Zaït, M. & Messatfa, H. (1997). A comparative study of clustering methods. *Future Generation Computer Systems*, vol. 13, pp. 149-159.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, vol. 8, no. 3, pp. 338-353.

Zeki, S. (1993). *A vision of the brain*. Blackwell Scientific Pub., Oxford, London.

Zrehen, S. (1993). Analysing Kohonen maps with geometry. In: Gielen, S. & Kappen, B. (Eds.). *Proc. of the Intl. Conf. on Artificial Neural Networks*, London: Springer.

Zupan, J. & Gasteiger, J. (1993). *Neural Networks for Chemists: An Introduction*. VCH Publishers, New York, NY.

Zurada, J. M. (1992). *Introduction to Artificial Neural Systems*. St. Paul - MN, West Publishing Company.

Índice de citações de autores

Chao & Stoner (1993): 111.
Clementine (1998): 100, 313.
Cole (1969): 19.
Cormack (1971): 15, 20.
Corral *et al.* (1994): 99.
Costa (1996a): 57, 65, 155, 290.
Costa (1996b): 57, 65.
Costa (1996c): 57.
Costa & Gonzaga (1996a): 57, 65.
Costa & Gonzaga (1996b): 57, 65.
Costa & Netto (1997): 57.
Costa *et al.* (1997a): 57.
Costa *et al.* (1997b): 150.
Costa & Netto (1998): 58, 182.
Costa & Netto (1999a): 58, 182.
Costa & Netto (1999b): 162.
Costa & Netto (1999c): 58, 99, 129, 197, 250, 252.
Cottrell *et al.* (1994): 80.
Cottrell *et al.* (1995): 98.
Cottrell (1997): 80, 98.
Cottrell *et al.* (1998): 98.
Darken & Moody (1991): 112.
Day & Edelsbrunner (1984): 20.
Debrun (1996): 6.
Debrun *et al.* (1996): 5.
Dedieu & Mazer (1992): 61.
Dempster *et al.* (1977): 46.
Diday & Simon (1978): 20.
Dijkstra *et al.* (1959): 150.
Dony & Haykin (1995): 99.
Dubes & Jain (1980): 20.
Duda & Hart (1973): 20, 56, 253.
Duda *et al.* (1998): 20, 44, 55, 57, 64, 171.
Duran & Odell (1974): 19, 26.
Durbin *et al.* (1989): 66.
Erwin *et al.* (1992a): 9, 79, 98.
Erwin *et al.* (1992b): 9, 79, 98.
Everitt (1974): 19.
Everitt (1981): 21.
Everitt (1993): 13, 15, 19, 20, 23, 30, 37, 38, 45.
Everitt & Hand (1981): 44.
Facon (1996): 150.
Favata & Walker (1991): 100.
Ferrán & Ferrara (1992): 100.
Firmin *et al.* (1997): 55, 112.
Fisher (1936): 253, 255.
Flanagan (1994): 80, 98.
Flanagan (1996): 80, 98.
Flexer (1997): 268, 312.
Flexer (1999): 100, 268, 312.
Florek *et al.* (1951): 32.
Fort & Pagès (1996): 98.
Frank *et al.* (1998):
Freeman (1991): 59.
Friedman & Rubin (1967): 38.
Fritzke (1993): 99, 123.
Fritzke (1994): 99, 123.
Fritzke (1995): 123.
Fritzke (1996): 123.
Fukushima (1975): 109.
Fukushima (1980): 109.
Fukushima *et al.* (1983): 109.
Fukushima (1988): 103, 110.
Fukushima (1991): 111.
Fukushima (1993): 109.
Gasteiger & Zupan (1993): 100.
Ghosh-Roy *et al.* (1998): 22.
Giardina & Dougherty (1988): 146.
Gonzales & Woods (1992): 142, 143, 144.
Goodacre (1994): 100.
Goodhill & Sejnowsky (1997): 277.
Gordon & Henderson (1977): 38.
Gordon (1981): 57, 204.
Gordon (1987): 20.

Li *et al.* (1993a): 115.
Lorr (1983): 19.
Lu & Shin (1992): 99.
Lund *et al.* (1998): 23.
MacQueen (1967): 21, 38.
Mandel & Chernyl (1988): 21.
Mao & Jain (1996): 263.
Mangiameli *et al.* (1996): 312.
Mardia *et al.* (1979): 26, 28, 57.
Maren *et al.* (1990): 63, 64.
Marr (1982): 144.
Marriott (1982): 38, 40.
Mascarenhas & Velasco (1989): 25, 29.
Massart & Kaufman (1983): 20.
Masters (1995): 62.
Matheron (1975): 146.
McLachlan & Basford (1988): 20, 44, 55.
Meijster & Roerdink (1996): 148, 150.
Merelo *et al.* (1994): 100.
Merkl (1997): 132, 133.
Merkl (1998): 315.
Merkl & Rauber (1997a): 132, 133.
Merkl & Rauber (1997b): 132, 133.
Meyer & Beucher (1990): 147, 150.
Meyer (1993): 147, 148.
Michaud (1997): 20, 36.
Miikulainen (1990): 127.
Miikulainen (1991): 127.
Miikulainen (1993a): 99, 127.
Miikkulainen (1993b):
Milligan (1980): 20.
Milligan & Cooper (1985): 20, 35, 56, 134.
Minnix *et al.* (1992): 111.
Missaoui & Frasson (1989): 22.
Mitchison (1995): 100.
Morasso & Sanguineti (1994): 62.
Morrison (1967): 28.
Murtagh (1983): 20.
Murtagh (1995): 134.
Murtagh (1996): 312.
Najman & Schmitt (1996):
Nevo *et al.* (1991): 23.
Nicolis & Prigogine (1989): 7.
Noguet *et al.* (1996): 150.
Obermayer *et al.* (1991): 62.
Obermayer *et al.* (1992): 62.
Obermayer & Blasdel (1997): 100.
Oja (1992): 99.
Oja (1995): 98.
Pal *et al.* (1992): 98.
Pal *et al.* (1993): 98.
Pal & Bezdek (1995):
Pal & Pal (1993): 145.
Pal, S. (1994): 43.
Pan & Chen (1992): 99.
Park (1994): 198.
Parker (1994): 290.
Parker (1997): 142, 144, 145.
Pedrycz (1990): 14.
Pessoa (1996): 5.
Pinkowski (1989): 21.
Pomierski *et al.* (1993): 62.
Popchev & Peneva (1988): 21.
Racz & Klotz (1991): 115, 116.
Ramussen (1990): 22.
Ramussen & Willett (1989): 22.
Rauber & Merkl (1999): 135.
Redner & Walker (1984): 46.
Richard & Lippman (1991): 62.
Ripley (1994): 62.
Ripley (1996): 62, 171, 199.
Rissanen (1978): 112, 114.
Rissanen (1989): 114.
Ritter (1989): 99.
Ritter & Kohonen (1989a): 249, 250, 252.
Ritter & Kohonen (1989b): 249, 250, 252.

Ritter *et al.* (1992): 97, 99.
Ritter & Schulten (1986): 98.
Ritter & Schulten (1988): 98.
Robbins & Monro (1951): 79.
Rumelhart & Zipser (1985): 66, 103.
Rumelhart *et al.* (1986): 62, 65.
Russ (1995): 142.
Sammon (1969): 268.
SAS (1996): 55.
Saxon (1991): 62.
Scholtes (1991): 100.
Scoltock (1982): 19.
Scott & Symons (1971): 21.
Serra (1982): 146.
Serrano *et al.* (1993): 100.
Shepard *et al.* (1962a): 268.
Shepard *et al.* (1962b): 268.
Siemon & Ultsch (1990): 100.
Sirosh & Miikkulainen (1993): 99.
Sneath (1957): 32.
Sokal & Sneath (1973): 19, 30.
Späth (1980): 20, 25, 28, 104.
Späth (1985): 25, 39.
Speckmann *et al.* (1994): 274.
Su *et al.* (1997): 216.
Sutton *et al.* (1994): 62.
Symons (1981): 21.
Tenenbaum *et al.* (1995): 199.
Thode *et al.* (1988): 21.
Timmins *et al.* (1999): 313.
Ting & Chuang (1993): 111.
Titterington *et al.* (1985): 20, 44, 55.
Tokutaka *et al.* (1999): 100.
Torgerson (1952): 268.
Tolat (1990): 98.
Tryon & Bailey (1970): 19.
Turner *et al.* (1993): 99.
Ultsch (1993a): 138.
Ultsch (1993b): 138.
Ultsch (1995): 162.
van Gils & Cluitsman (1993): 62
Villmann *et al.* (1994): 275, 276.
Villmann *et al.* (1997): 270, 273.
Visa (1990a): 99.
Visa (1990b): 99.
Visa (1992): 99.
Visa (1994): 99.
von der Malsburg (1973): 61.
Waller *et al.* (1998): 312.
Walter & Schulten (1993): 99.
Wan & Fraser (1993): 99, 126.
Wan & Fraser (1994a): 99, 126.
Wan & Fraser (1994b): 126.
Wan & Fraser (1994c): 126.
Ward (1963): 34.
Weinstein *et al.* (1995): 100.
Whittington & Spracklen (1994): 100.
Willshaw & von der Malsburg (1976): 61.
Wilson (1994): 100.
Wiskott & von der Malsburg (1996): 104, 119.
Wolfe (1970): 21, 55.
Wolfe (1978): 55.
Wong & Lane (1983): 35.
Wyler (1993): 100.
Yin & Allinson (1993): 98.
Yovits & Cameron (1960): 61.
Zaït & Messatfa (1997): 20.
Zadeh (1965): 40.
Zeki (1993): 8, 60, 65.
Zrehen (1993): 277.
Zupan & Gasteiger (1993): 88, 97, 100, 158.
Zurada (1992): 104.