INTELIGÊNCIA COMPUTACIONAL NA CLASSIFICAÇÃO LITOLÓGICA

Daniel Rodrigues de Silos Moraes

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA CIVIL.

Aprovada por:

_____

Prof. Alexandre Gonçalves Evsukoff, Dr.

_____

Prof. Nelson Francisco Favilla Ebecken , D.Sc.

_____

Prof. Luiz Pereira Calôba, Dr.Ing.

_____

Dr. Sebastião Cesar Assis Pereira, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

OUTUBRO DE 2004

MORAES, DANIEL RODRIGUES DE SILOS

Inteligência Computacional na Classificação Litológica [Rio de Janeiro] 2004

vii, 75p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia Civil., 2004)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Redes neurais,

Classificação não-supervisionada,

Classificação litológica

I. COPPE/UFRJ II. Título (série)

Agradecimentos

A Deus, pela oportunidade de viver e realizar este trabalho.

Ao meu pai (in memorian) pelo seu exemplo acadêmico e profissional.

À minha mãe, pelo seu profundo amor.

À minha mulher, por seu amor e compreensão pelo tempo em que me dediquei a este trabalho.

Aos meus avós Aluysio e Nize pelo amor e auxílio dispensados durante toda minha vida e em especial a este mestrado.

Aos amigos José Luiz dos Anjos Rosa, Myrian Costa, Leonardo Valente e Ângelo Silva pelo auxílio em vários momentos.

À amiga Márcia Kuhn e a todos os demais amigos do Lab2M, pelo auxílio e conhecimento imprescindíveis a este trabalho.

À CAPES, pelo financiamento da pesquisa elaborada nesta tese.

À ANP, pela liberação da base de dados utilizada nesta tese.

Aos meus orientadores Profs. Alexandre G. Evsukoff e Nelson F. F. Ebecken, pela confiança depositada e pelo auxílio em momentos de dificuldade.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

INTELIGÊNCIA COMPUTACIONAL NA CLASSIFICAÇÃO LITOLÓGICA

Daniel Rodrigues de Silos Moraes

Outubro/2004

Orientadores: Alexandre Gonçalvez Evsukoff
Nelson Francisco Favilla Ebecken

Programa: Engenharia Civil

Métodos de agrupamento de dados são ferramentas importantes em mineração de dados, pois permitem detectar padrões não encontrados explicitamente nos dados armazenados. Neste trabalho, métodos de agrupamento são aplicados a atributos sísmicos visando detectar padrões que permitam identificar diferentes litologias na análise exploratória de petróleo.

Atributos sísmicos são gerados a partir de transformações sobre os dados de uma linha sísmica, sendo alguns selecionados para formar um conjunto de dados a ser aplicado pelos diferentes algoritmos de agrupamento de dados.

Este trabalho apresenta a utilização de quatro métodos de agrupamento de dados diferentes: As redes neurais conhecidas por Competitiva, SOM (*Self-Organizing Maps"*) e UVQ (*"Unsupervised Vector Quantiser"*) e o algoritmo baseado em teoria de conjuntos nebulosos conhecido por FCM (*"Fuzzy C-Means"*).

Na aplicação de cada método, várias partições são geradas e é aplicada uma função de validação de agrupamento de dados conhecida por índice PBM. Esta função indica o número do particionamento em que os dados foram melhor agrupados e os resultados são apresentados e analisados.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

COMPUTING INTELLIGENCE IN LITOLOGY CLASSIFICATION

Daniel Rodrigues de Silos Moraes

October/2004

Advisors: Alexandre Gonçalvez Evsukoff
Nelson Francisco Favilla Ebecken

Department: Civil Engineering

Clustering methods are important tools in data mining, allowing the detection of hidden patterns that are not explicitly recorded in data. Their applications to seismic attributes have the aim of detecting seismic patterns that would enable the identification of lithologies in exploration analysis of Petroleum.

Seismic attributes are generated from transformation of the data from a seismic line. Some attributes are selected in order to form a data set that is used by the clustering methods.

This work presents an application using four distinct clustering methods: artificial neural networks known as Competitive, SOM ("Self-Organizing Maps") and UVQ ("Unsupervised Vector Quantiser") and the algorithm based on fuzzy sets known as FCM ("Fuzzy C-Means").

In the application of each method, several partitions are generated and a cluster validity function, called PBM index, is applied. This function shows the partition number corresponding to the best clustering and the results are presented and analyzed.

Sumário

1 Introdução **1**
1.1 Objetivo 4
1.2 Organização do Trabalho 5

2 Metodologia **6**
2.1 Redes Neurais 8
2.2 Redes Neurais Baseadas em Competição 9
2.3 Rede Competitiva 10
2.3.1 Arquitetura 10
2.4 Rede SOM 12
2.4.1 Arquitetura 15
2.5 Rede UVQ 16
2.5.1 Arquitetura 16
2.6 Algoritmo FCM 18
2.7 Validação de Cluster 20
2.8 Índice PBM 21

3 Aplicação **24**
3.1 A Base de Dados 24
3.2 Programas Utilizados 25
3.2.1 OpendTect 25
3.2.2 Matlab e SOM_PAK 35
3.2.2.1 Importa_janela 35
3.2.2.2 Agrup_inicio 38
3.2.2.3 Agrup_comp e Agrup_fcm 39
3.2.2.4 Aplica_validação e Aplica_val_som 40

4 Resultados **41**
4.1 Apresentação dos Resultados 41
4.1.1 Detalhes sobre os conjuntos de dados 41
4.1.2 Resultados 43
4.1.3 Análise dos Resultados 47

5 Conclusões e perspectiva de trabalhos futuros **54**
5.1 Perspectiva de trabalhos futuros 55

Apêndices **57**

A Rotinas e funções utilizadas na fase de aplicação e validação de agrupamentos **57**

Referências Bibliográficas **74**

Capítulo 1

Introdução

O Petróleo é umas das principais fontes de energia da atualidade. Várias técnicas são empregadas com o intuito de se descobrir o petróleo antes de se perfurar um poço, mas nenhuma delas consegue detectar uma jazida de petróleo em uma razão de 100% de acerto. Dificilmente o petróleo é encontrado na primeira perfuração de um poço em uma área ainda não explorada. As técnicas empregadas nas fases que antecedem a perfuração de um poço identificam áreas (rochas) que são propícias à acumulação de hidrocarbonetos [1]. Elas são aplicações de conhecimentos extraídos das ciências Geologia e Geofísica. Este trabalho concentra-se nas fases de prospecção do petróleo e tem sua aplicação relacionada com estas duas ciências. Conhecimentos sobre a geologia do petróleo, sua formação, migração e aprisionamento são indispensáveis ao entendimento deste trabalho e podem ser encontradas em [1, 2, 3, 4].

Métodos geológicos e geofísicos são aplicados com o objetivo de se obter dados sobre a subsuperfície (interior da terra). Segundo THOMAS [1], o método sísmico de reflexão é o mais utilizado na indústria do petróleo atualmente, representando cerca de 90% dos investimentos em prospecção. Seu diferencial encontra-se no fornecimento de alta definição das feições geológicas em subsuperfície que são propícias à acumulação de hidrocarbonetos, a um custo relativamente baixo. Por esta razão, este trabalho utilizará dados sísmicos em sua aplicação. A expressão "dados sísmicos" refere-se a um conjunto de dados resultantes da aplicação do método sísmico de reflexão, colhidos através de sensores que são sensíveis às vibrações de pressão nas partículas da superfície d'água, quando estimuladas por ondas sísmicas específicas [5]. THOMAS [1] apresenta uma introdução sobre o método sísmico de

reflexão.

Manipulando-se dados sísmicos é possível gerar transformadas [5] ou atributos sísmicos. TANER [6] define atributos sísmicos como todas as informações provenientes de dados sísmicos, sejam através de medidas diretas, lógicas ou experiências baseadas no raciocínio. Informações detalhadas sobre cálculos e características de atributos sísmicos são encontradas em [5, 6, 7]. Atributos sísmicos enriquecem a análise de dados sobre a subsuperfície destacando características diferentes. Por exemplo: o atributo amplitude instantânea mede a intensidade da refletividade e está associada à energia do sinal sísmico. O atributo fase instantânea é uma medida de continuidade de um evento. O atributo freqüência instantânea expressa a razão de mudança no tempo do atributo fase [5].

Após a análise de atributos sísmicos é decidido o local da perfuração de um poço. Esta tarefa é a que demanda mais recursos financeiros na fase de prospecção. Nem sempre a perfuração de um poço é considerada bem-sucedida. O petróleo pode não ser encontrado ou o poço pode ser considerado não-comercialmente viável. Após a abertura (perfuração) de alguns poços, decidir onde perfurar novos poços torna-se mais fácil, graças ao conhecimento obtido da área em torno das perfurações já ocorridas. Este conhecimento, em parte, é adquirido através da aplicação de outras técnicas realizadas durante ou após a perfuração. O conhecimento de uma área estudada é acrescido através da abertura de mais poços e como conseqüência, pode haver um acréscimo na certeza da localização de uma jazida de petróleo e na decisão onde novos poços devem ser abertos.

Como a perfuração é um processo que representa um alto custo, várias pesquisas vêm sendo empregadas com o objetivo de diminuir o risco da perfuração de um poço. Um grande desafio a ser vencido é encontrar petróleo perfurando-se um mínimo de poços possíveis, diminuindo-se o alto custo da perfuração. Em outras palavras, deseja-se aumentar a certeza da localização do petróleo antes de se perfurar, diminuindo-se o risco exploratório.

As pesquisas em torno da subsuperfície continuam durante e após a perfuração. Duas técnicas são utilizadas na identificação de litologias: A Testemunhagem e a Perfilagem.

A Testemunhagem é realizada através do recolhimento de fragmentos de rochas no momento da perfuração de um poço. Esta técnica demanda um alto custo financeiro. Além de ser necessário interromper a perfuração para que fragmentos de rochas possam ser recolhidos, especialistas como geólogos e paleontólogos são necessários para analisar os fragmentos e determinar informações como idade e características das rochas.

A perfilagem demanda recursos bem menos dispendiosos quando comparados aos recursos aplicados à testemunhagem. Não há necessidade de se interromper o processo de perfuração para aplicá-la. O perfil de um poço é uma imagem visual, em relação à profundidade, de uma ou mais características ou propriedades das rochas perfuradas [1]. Resistividade elétrica, potencial eletroquímico natural, tempo de trânsito de ondas mecânicas, radioatividade natural ou induzida são exemplos de perfis que são obtidos através do deslocamento contínuo de um sensor de perfilagem (sonda) dentro do poço. Genericamente são chamados de perfis elétricos, independentemente do processo físico de medição utilizado [1]. O perfil de raios gama (GR do inglês *"Gama Ray"*) é utilizado na detecção da radioatividade total da formação geológica, na identificação de litologia e minerais radioativos e no cálculo do volume de argilas ou argilosidade [1]. Neste trabalho, este perfil foi utilizado para identificar pacotes de areias na sísmica.

A necessidade de se obter conhecimentos em áreas distintas torna as pesquisas em hidrocarbonetos um meio interdisciplinar. Possivelmente, a resposta para um acréscimo na certeza da identificação de novas jazidas bem como um decréscimo no tempo de análise de uma bacia ou reservatório está em incluir técnicas de mineração de dados. Estas técnicas podem tornar as tarefas acima menos custosas. Elas têm sido amplamente utilizadas na indústria do petróleo. O uso de algoritmos de Inteligência computacional como Redes Neurais Artificiais, Lógica Nebulosa (*"Fuzzy Logic"*), Algoritmos Genéticos e algoritmos híbridos vêm sendo utilizados em caracterização de reservatórios, classificação e previsão de porosidade, classificação não-supervisionada entre dados sísmicos e perfis de poços, etc.

Neste trabalho, algoritmos são utilizados em uma abordagem de aprendizado não-supervisionado objetivando-se agrupar os dados (atributos sísmicos) em diferentes agrupamentos (clusters). Três modelos de redes neurais e um modelo de lógica nebulosa são utilizados. Os dados são separados por similaridade entre eles com o objetivo de identificar padrões sísmicos que separem diferentes litologias.

Dados de perfilagem foram utilizados apenas na identificação de pacotes de areias através do conhecimento de especialista em geologia. Estes não foram utilizados no processamento, apesar de existirem abordagens em que os dados sísmicos e perfis de poços o são.

As areias possuem propriedades que as caracterizam como possíveis reservatórios de petróleo. Segundo o conhecimento de especialista, na área estudada existe o interesse de se descobrir se as areias detectadas são turbidíticas, pois cerca de 90% dos reservatórios

de petróleo no mundo estão localizados em areias turbidíticas. Na Bacia de Campos, por exemplo, 83% dos reservatórios estão associados aos depósitos turbidíticos [8].

A ANP (Agência Nacional de Petróleo) cedeu um conjunto de dados à Universidade Federal do Rio de Janeiro. Este conjunto contém cinco linhas sísmicas e três registros de poços. A existência de poucos registros de poços torna inadequado o processamento integrado entre dados sísmicos e de perfis, por haver a necessidade de mais dados de poços ao longo de uma ou mais linhas sísmicas para que eles pudessem vir a ser utilizados no processamento. Apenas os dados sísmicos (atributos sísmicos) foram utilizados no processamento. É importante destacar que o enfoque deste trabalho não é identificar litologias (tipos de rochas), cuja tarefa é de competência de especialistas (geólogo e geofísico).

1.1 Objetivo

O Objetivo deste trabalho é analisar o desempenho de métodos de agrupamento de dados sobre uma base de dados composta por diferentes atributos sísmicos. Esta avaliação é realizada através da aplicação do índice de validação de cluster conhecido por PBM, cujo nome é formado pelas iniciais dos sobrenomes dos autores (Pakhira, Bandyopadhyay e Maulik) [9]. O índice é aplicado aos resultados dos diferentes algoritmos de agrupamentos, sendo que em cada algoritmo é determinado um número inicial e um número final de agrupamentos. Em outras palavras para cada algoritmo de agrupamento, a base de dados é separada, por exemplo, em cinco classes, em seguida em seis, sete, etc. até um valor máximo, sempre incrementando-se em um o número de agrupamentos. O maior índice encontrado corresponde à melhor configuração que é composta pelo algoritmo de agrupamento e um determinado número de agrupamentos.

A partir da identificação dos melhores resultados obtidos, especialistas poderiam utilizá-los na tentativa de identificar litologias e possíveis jazidas de petróleo, pois os melhores resultados assegurarão a existência de um possível padrão sísmico que separa diferentes litologias.

A identificação dos pacotes de areia são importantes na visualização dos resultados obtidos pelos diferentes algoritmos de agrupamento de dados, sendo possível identificar se há um padrão que separe estas áreas das demais. Ocorrendo esta separação, pode-se afirmar que possivelmente há áreas que correspondam a areias, mas somente o conhecimento de

especialistas pode confirmar tais suspeitas, como descrito anteriormente.

1.2 Organização do Trabalho

O restante desta tese está organizado como se segue:

No Capítulo 2 são apresentados os métodos de agrupamento de dados utilizados e o índice de validação PBM. Inicialmente é apresentada uma introdução sobre agrupamento de dados, seguida por uma introdução sobre redes neurais artificiais incluindo detalhes dos modelos de redes neurais que foram utilizados na fase de aplicação da tese. O algoritmo FCM (*"fuzzy c-means"*) é também apresentado. Detalhes sobre o índice de validação PBM é apresentado no final deste capítulo.

No capítulo 3 são apresentados a base de dados e os programas utilizados neste trabalho, incluindo os programas que utilizam os métodos de agrupamentos apresentados no capítulo 2. Inicialmente são descritos detalhes sobre a base de dados. Em seguida, são apresentados os programas OpendTect, SOM_PAK e Matlab (incluindo o conjunto de rotinas e funções conhecidos por Segy-MAT). Finalmente, são apresentados descrições sobre rotinas (*"scripts"*) e funções criadas no programa Matlab. Estas permitirão a realização de aplicações de diferentes modelos de agrupamento de dados em diferentes programas (OpendTect e Matlab).

No Capítulo 4 são apresentados os resultados da aplicação do índice de validação de cluster PBM e uma análise dos melhores resultados obtidos.

No Capítulo 5 é apresentada a conclusão deste trabalho.

O Apêndice A contém os códigos-fonte de rotinas e funções criados no programa Matlab.

Capítulo 2

Metodologia

Quanto ao processo de aprendizagem, os métodos de mineração de dados podem ser divididos em dois principais grupos: aprendizagem supervisionada e não-supervisionada.

Na aprendizagem supervisionada, há uma relação entre as observações e os agrupamentos de dados, ou seja, são conhecidas as classes pelas quais o conjunto de dados é dividido. Em outras palavras, a aprendizagem supervisionada é caracterizada pela existência de um professor. A idéia é ensinar ao método um padrão conhecido, fornecendo um conjunto de dados de entrada e saída. O Conjunto de dados é dividido em duas partes. A primeira é chamada de conjunto de treinamento, pois será utilizada na fase de treinamento. Além do vetor de entrada, um vetor de saída é também informado e o método atualizará vetores de pesos na tentativa de aprender padrões existente entre os dados de entrada e os dados de saída. Em outras palavras, o método deverá aprender que para um determinado vetor de entrada ele deverá produzir uma saída igual à informada pelo professor. Ao final, ele deverá gerar resultados que satisfaçam uma aplicação específica (previsão, classificação, etc.). Isto ocorrerá caso ele apresente uma porcentagem de acerto considerado aceitável pelo seu usuário.

Após a fase de treinamento , um conjunto de dados denominado conjunto de teste é utilizado a fim de testar o aprendizado do algoritmo. Nesta fase não é fornecida nenhuma saída, mas após a produção da respostas, as saídas são novamente comparadas a fim de identificar se o algoritmo conseguiu aprender um padrão que sirva não só para o conjunto de dados utilizado na fase de treinamento, mas também para o conjunto de dados de teste, ou seja, o conjunto de dados utilizado na fase de teste tem o objetivo de testar a aprendizagem do método de agrupamento de dados.

Informações sobre métodos que seguem esta abordagem de aprendizagem podem ser encontradas em [10, 11, 12, 13]. Aplicação de redes neurais com aprendizagem supervisionada na indústria de petróleo podem ser encontradas em [14, 15].

Clusterização ou Agrupamento de Dados é uma das tarefas mais usuais em processos de mineração de dados na descoberta e identificação de distribuições e padrões de interesse. Através da aplicação de um método de agrupamento de dados é possível agrupar um conjunto de dados em classes de elementos similares. Estes métodos são caracterizados pela aplicação de algoritmos de classificação não-supervisionada.

A classificação não-supervisionada consiste em separar um conjunto de dados utilizando-se apenas informações provenientes dos atributos que compõem o conjunto de dados. O conjunto de treinamento $T = \{(\mathbf{x}(t)), t = 1..n\}$ contém assim, em cada registro, somente a informação do vetor de atributos $\mathbf{x}(t) = (x_1(t), ..., x_p(t)) \in \mathbb{R}^p$. Em outras palavras, a aprendizagem não-supervisionada é caracterizada pela ausência de um professor, ou seja, não há um conjunto de dados de saída fornecido, existindo apenas um conjunto de dados de entrada.

Um método de agrupamento de dados separa um conjunto de dados em um determinado número de grupos, definido pelo usuário. Os dados são separados em grupos por similaridade entre eles. Tais dados podem representar a existência de padrões não naturalmente detectáveis, isto é, informações e conhecimentos existentes e que permanecem ocultos na maneira tradicional em que os dados são armazenados podem ser descobertos aplicando-se técnicas de agrupamento de dados.

Existe uma variedade de métodos de agrupamento de dados desenvolvidos a partir de técnicas estatísticas, redes, neurais, conjuntos nebulosos e algoritmos genéticos. Este trabalho apresenta métodos de redes neurais e conjuntos nebulosos que realizam o particionamento, ou seja, separam um conjunto de dados em K grupos, encontrando a matriz de coordenadas dos centros de agrupamento $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_K]$ onde cada coluna $\mathbf{w}_i \in \mathbb{R}^p$ define as coordenadas do centro do agrupamento representativo da classe $\mathcal{C}_i$, sendo $i = 1, ..., K$.

Os métodos de particionamento visam a otimização de um critério de custo definido em função da distância dos registros do conjunto de treinamento em relação aos centros dos agrupamentos (equação 2.1).

$$J(w) = \frac{1}{n}\sum_{t=1}^{n}\sum_{\mathbf{x}(t)\in\mathcal{C}_i} d(\mathbf{x}(t), \mathbf{w}_i)^2 \tag{2.1}$$

onde $d(\mathbf{x}(t), \mathbf{w}_i)$ é geralmente, a distância Euclideana, embora outras métricas de distância possam ser utilizadas. Genericamente, a distância euclideana pode ser calculada como (equação 2.2):

$$d(\mathbf{x}(t), \mathbf{w}_i)^2 = [\mathbf{x}(t) - \mathbf{w}_i]\mathbf{P}[\mathbf{x}(t) - \mathbf{w}_i]^T \tag{2.2}$$

onde a matriz de ponderação **P**, que define os pesos de uns atributos em relação aos outros. Na falta de maiores informações adota-se a matriz de identidade $\mathbf{I}^p$.

Os métodos baseados em distância são afetados pela diferença de escala entre os valores dos atributos, sendo necessário normalizar os atributos no intervalo [0,1] pelos valores máximo e mínimo do conjunto de treinamento. Esta normalização pode ser realizada através da seguinte expressão:

$$\mathbf{x}^{norm}(t) = \frac{\mathbf{x}(t) - \mathbf{min}(x)}{\mathbf{max}(x) - \mathbf{min}(x)} \tag{2.3}$$

onde $\mathbf{x}^{norm}$ é o vetor de atributos normalizados, $\mathbf{max}(x)$ é o vetor com os valores máximos das coordenadas de cada atributo e $\mathbf{min}(x)$ o vetor com os valores mínimos das coordenadas de cada atributo. As próximas seções apresentam os métodos de agrupamento de dados utilizados na aplicação deste trabalho.

2.1 Redes Neurais

Redes Neurais Artificiais ou simplesmente, Redes Neurais são modelos de computação caracterizada por sistemas que relembram (em algum nível) a estrutura do cérebro humano. Não é baseada por regras ou programas e por isso é uma alternativa à computação algorítmica convencional [10]. De acordo com HAYKIN [11]: "Uma rede neural é um processador paralelamente distribuído constituído de unidades de processamento simples, que tem a propensão natural de armazenar conhecimento experimental e torná-lo disponível para o uso. Sua semelhança com o cérebro se deve por adquirir conhecimento por meio de processos de aprendizado e pelas "forças" de interconexão entre os neurônios, conhecidas como pesos

sinápticos e que são usadas para armazenar o conhecimento adquirido".

As redes neurais são reconhecidas como ferramentas importantes com relação à não-linearidade [10], característica esta bastante evidente em atributos sísmicos [5].

Portanto, o uso de redes neurais como ferramentas de agrupamento de dados tem tido expressivo sucesso na Indústria de Petróleo [5, 14].

Em seguida é apresentada uma teoria baseada nas redes-neurais com aprendizagem não-supervisionada conhecidas também como redes neurais baseadas em competição.

2.2 Redes Neurais Baseadas em Competição

Há ocasiões em que não há dados cujo padrão é conhecido. Em outras palavras, não são conhecidas as classes do conjunto de dados. Conseqüentemente, não é possível utilizar uma rede neural supervisionada, pois não existe a possibilidade de se utilizar um professor que ensine à rede um padrão existente.

Para resolver este problema, utiliza-se uma forma de rede neural cuja aprendizagem é não-supervisionada e possui um mecanismo conhecido por competição [13].

A forma de competição mais conhecida é a chamada "O vencedor leva tudo" (do inglês *"Winner takes all"*) [10, 11, 12, 13], onde um grupo de neurônios competem entre si e somente um será o vencedor.

Esta forma representa um comportamento baseado no cérebro humano quando este necessita tomar uma decisão. Suponha que um professor receba um convite para lecionar três cursos no mesmo dia e horário. É impossível que o professor aceite dois ou os três convites, podendo apenas aceitar um. Logo, o professor deve escolher somente uma das três opções acima. Uma rede baseada em competição funciona de modo semelhante. Havendo três neurônios na camada de competição, apenas um será "ligado" (representando uma opção), enquanto os demais permanecerão "desligados".

O número de neurônios que irão competir entre si representa o número de agrupamentos que se deseja formar. Durante o treinamento, a rede determina a unidade de saída (neurônio vencedor). O vetor de pesos (sinapses) é então ajustado de acordo com um algoritmo de aprendizado da rede.

Vários modelos de redes neurais utilizam um algoritmo de aprendizado conhecido por regra ou aprendizagem de Kohonen [13]. Nesta forma de aprendizado, o vetor de pesos do

neurônio vencedor terá seus pesos atualizados, isto é, será formado um novo vetor de pesos que é uma combinação linear do vetor de pesos antigo e o vetor de pesos atual.

A atualização do pesos de um neurônio vencedor v é dada como

$$\mathbf{w}_v(q+1) = \mathbf{w}_v(q) + \alpha[\mathbf{x} - \mathbf{w}_v(q)] \tag{2.4}$$

ou

$$\mathbf{w}_v(q+1) = \alpha\mathbf{x} + (1-\alpha)\mathbf{w}_v(q) \tag{2.5}$$

onde q é o valor da iteração, $\mathbf{x}$ é o vetor de entrada, $\mathbf{w}_v$ é o vetor de pesos associado ao neurônio vencedor v e α, a taxa de aprendizagem [13].

Cada vez que um vetor de entrada é apresentado à rede, esta mede a distância euclidiana (equação 2.2) entre este vetor e cada um dos vetores de peso. O neurônio cujo vetor de pesos mais se aproxima do vetor de entrada ganha a competição. Todos os métodos de agrupamento de dados deste trabalho utilizam esta métrica de distância.

Três diferentes modelos de redes neurais não-supervisionadas, que possuem semelhanças e algumas diferenças entre si foram utilizados neste trabalho. Um modelo, conhecido como rede competitiva faz parte da caixa de ferramentas ("*toolboxes*") do programa Matlab. A rede SOM faz parte de um conjunto de rotinas e funções do programa SOM_PAK [16]. O terceiro modelo, conhecido como UVQ faz parte do programa OpendTect [17]. Uma introdução à teoria destes modelos é apresentada a seguir.

2.3 Rede Competitiva

A rede competitiva, também conhecida como camada de Kohonen ou camada competitiva de Kohonen possui duas camadas: uma de **Entrada** e outra conhecida por **Camada Competitiva**.

A seguir são apresentados detalhes sobre a arquitetura deste modelo:

2.3.1 Arquitetura

A figura 2.1 exibe a arquitetura de uma rede competitiva.

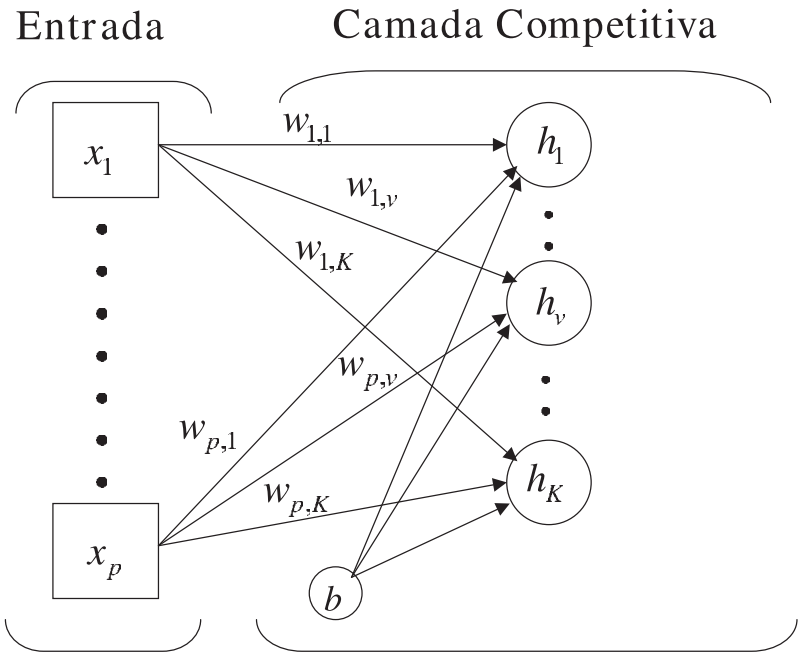


FIGURA 2.1: Arquitetura da Rede Competitiva.

O vetor **x** representa um vetor do conjunto de dados que é apresentado à rede. Os vetores **w** representam os pesos dos neurônios e h, os neurônios da camada competitiva. Tradicionalmente, este modelo calcula a distância euclidiana entre o vetor de entrada e cada vetor de peso da rede. O neurônio da camada competitiva que possui o vetor de peso cuja distância do vetor de entrada foi menor, dada por 2.2, ganha a competição. Na figura 2.1 ele é representado por h_v. Este neurônio, portanto recebe o valor 1 enquanto os demais permanecem com o valor 0. O neurônio vencedor é atualizado através da regra de aprendizagem de Kohonen (equação 2.5).

Uma variação deste modelo tradicional utiliza um polarizador ("*bias*"), representado por b que é somado ao resultado do cálculo da distância euclidiana entre o vetor de entrada e um vetor de pesos.

Uma das limitações das redes competitivas é que alguns neurônios podem não ser sempre alocados. Em outras palavras, alguns vetores de pesos podem ser iniciados longe de qualquer vetor de entradas e jamais ganhar uma competição, não importando o quanto a fase de treinamento dure. O resultado é que seus pesos não conseguem armazenar conhecimento e eles nunca vencem uma competição. Estes neurônios, chamados de "neurônios mortos" não possuem utilidade.

Na tentativa de reverter esta situação, polarizadores são utilizados para dar uma vantagem aos neurônios que nunca ou raramente ganham uma competição sobre os que freqüentemente ganham. Um polarizador altera a distância de um neurônio tornando-o mais suscetível a vitória.

Primeiramente, uma média das saídas dos neurônios é calculada. Isto é equivalente a porcentagem de vezes que cada neurônio vence uma competição. Esta média é utilizada para atualizar os polarizadores com a função de aprendizagem, tornando os polarizadores dos neurônios freqüentemente ativos menores, e os polarizadores dos neurônios menos ativos, maiores. Dessa maneira, é possível tentar colocar os neurônios iniciados longe dos vetores de entrada, em uma posição mais próxima, permitindo assim, que eles passem a vencer competições e que os dados venham a ser agrupados em um número de grupos idêntico ao número de neurônios. Este mecanismo é também conhecido por consciência [13].

2.4 Rede SOM

A rede SOM, cuja sigla significa Mapas Auto-Organizáveis (do inglês *"Self-Organizing Maps"*) é o modelo de redes neurais mais famoso, dentre os propostos por Teuvo Kohonen [16].

A principal diferença entre a rede SOM e a competitiva está no conceito de vizinhança. A rede SOM aprende a reconhecer seções de vizinhanças no espaço de entrada. Ela aprende a distribuição (como ocorre na rede competitiva) e a topologia dos vetores de entrada sobre os quais ocorre o treinamento.

Os neurônios na camada de competição de uma rede SOM são arranjados originalmente em posições físicas de acordo com a função de topologia. Pode-se arranjar os neurônios em topologias em forma de grade (retangular) ou hexagonal.

Uma rede SOM identifica um neurônio vencedor usando o mesmo procedimento empre-

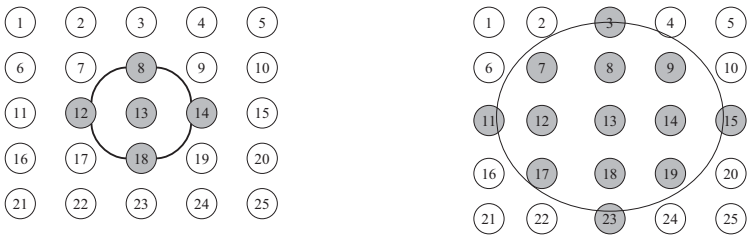


FIGURA 2.2: Vizinhança de um neurônio. A figura à esquerda exibe a vizinhança de raio $r = 1$ e a da direita, vizinhança de raio $r = 2$

gado em uma rede competitiva. No entanto, em vez de atualizar apenas o neurônio vencedor, todos os neurônios a uma certa vizinhança do neurônio vencedor são atualizados utilizando-se a regra de Kohonen.

Dessa forma, quando um vetor de entrada **x** é apresentado, os pesos do neurônio vencedor e seus vizinhos mais próximos movem-se em sua direção. Por conseguinte, após muitas apresentações, os neurônios vizinhos irão aprender vetores similares uns aos outros. Para ilustrar o conceito de vizinhança, considere a figura 2.2.

O diagrama à esquerda mostra uma vizinhança bidimensional de raio $r = 1$ em torno do neurônio 13. O diagrama à direita mostra uma vizinhança de raio $r = 2$. Tais vizinhanças podem ser escritas como se segue:

$N13(1) = \{8, 12, 13, 14, 18\}$ e

$N13(2) = \{3, 7, 8, 9, 11, 12, 13, 14, 15, 17, 18, 19, 23\}$

O arranjo de neurônios de uma rede SOM não precisa seguir um padrão bidimensional, necessariamente. Pode-se utilizar um arranjo unidimensional cujo neurônio tem somente dois vizinhos com um raio $r = 1$ (ou um vizinho singular se o neurônio encontra-se no final da linha). Pode-se também definir distâncias de diferentes maneiras, por exemplo, utilizando-se um arranjo retangular ou hexadecimal de neurônios e vizinhos. O desempenho

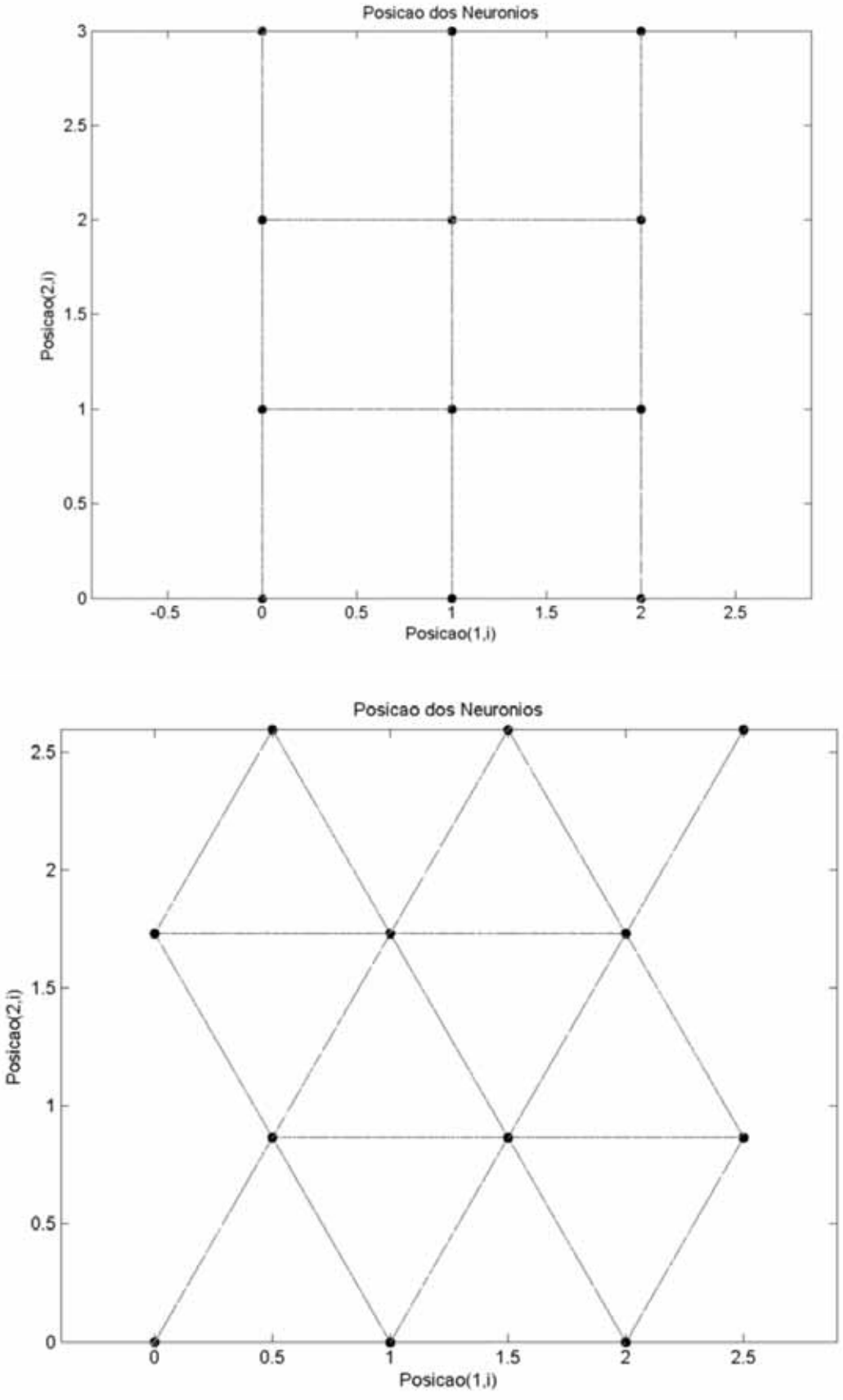


FIGURA 2.3: Topologias retangular e hexagonal

da rede não é sensitivo exatamente à forma de vizinhança.

A figura 2.3 mostra como um arranjo de 3 por 4 neurônios é organizado nas topologias retangular e hexagonal.

2.4.1 Arquitetura

A figura 2.4 mostra a arquitetura de uma rede SOM.

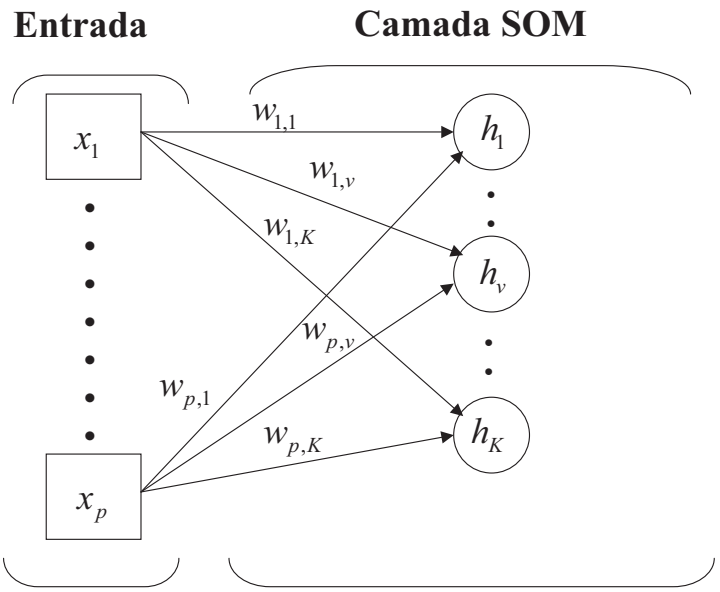


FIGURA 2.4: Arquitetura - Rede SOM.

Esta arquitetura é semelhante à arquitetura da rede competitiva, com exceção do polarizador que não é utilizado na rede SOM.

2.5 Rede UVQ

A rede UVQ, Quantização Vetorial Não-Supervisionada (do inglês *"Unsupervised Vector Quantiser"*), é uma versão modificada da rede LVQ (*"Learning Vector Quantiser"*) [10, 12]. Quantização Vetorial por Aprendizagem (LVQ) é uma importante aplicação de aprendizado competitivo largamente utilizada em dados codificados e dados de compressão [18].

Na Quantização Vetorial, um vetor de entrada é substituído pelo índice da unidade de saída vencedora. Agrupamentos são encontrados através do cálculo da distância euclideana (equação 2.2), como ocorre na rede competitiva.

A UVQ é similar à LVQ mas existem diferenças relacionadas a inicialização de pesos, que na UVQ é aleatória. Já o processo de competição e atualização dos pesos é idêntico aos da rede competitiva, ou seja, a cada vetor de entrada apresentado o neurônio da camada intermediária que se encontra mais próximo dele é atualizado, em direção ao próprio vetor de entrada [18].

2.5.1 Arquitetura

A figura 2.5 exibe detalhes sobre a arquitetura da rede UVQ.

Observa-se na figura 2.5 que a rede UVQ, diferentemente das outras redes já descritas, possui duas saídas. A primeira, representada por o_1 recebe o índice do neurônio vencedor. Suponha que uma configuração desta rede utilize três neurônios na camada intermediária (camada UVQ). Cada neurônio pode possuir dois valores: 0 ou 1, mas há apenas três possíveis combinações na formação de um conjunto de três neurônios. Uma vez que apenas um valor 1 é selecionado a cada vez, as possíveis são: [1 0 0], [0 1 0] ou [0 0 1]. Caso a primeira combinação ocorra, a primeira saída da rede UVQ será igual 1. Caso ocorra a segunda combinação, esta saída será igual a 2 e a terceira, igual a 3. Portanto, esta saída da rede refere-se ao índice do neurônio ativado (valor igual a 1). Logo, para uma combinação igual a [0 0 1], esta saída é igual a 3, pois o número 1 encontra-se no terceiro índice, representando a vitória da competição pelo terceiro neurônio. Este índice representa um agrupamento (classe).

A segunda saída (o_2) representa um grau de competição (*"match degree"*), isto é, uma medida de confiança de quão próximo o vetor de entrada encontra-se de seu centro de classe. O grau é apresentado através de um valor entre 0 e 1. Quanto mais próximo de 1, mais próximo o vetor de entrada encontra-se de seu centro de classe. Esta é uma característica importante,

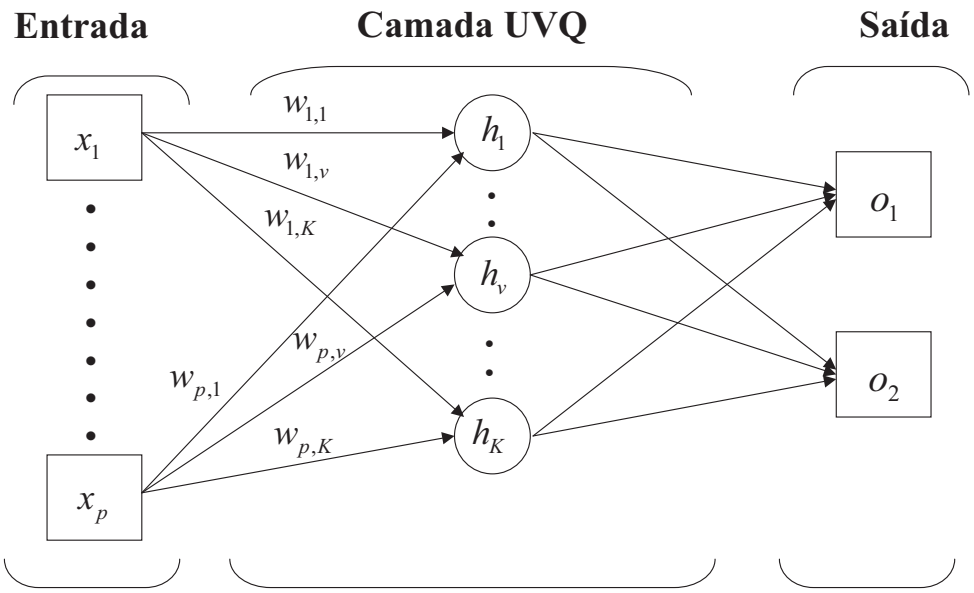


FIGURA 2.5: Arquitetura da Rede UVQ.

pois é uma informação que exibe a "qualidade" de uma classificação não-supervisionada. Caso uma média desta medida seja considerada baixa, pode-se afirmar que não houve um bom agrupamento de dados.

O grau de competição o_2 pode ser calculado como:

$$o_2 = (1 - \frac{1 - h_v(\mathbf{x})}{g\sqrt{p}}) \tag{2.6}$$

onde h_v representa o neurônio vencedor, **x** o vetor de entrada, g é o intervalo da variação para os dados de treinamento e p o número de atributos do vetor de entrada [17].

No OpendTect, a rede UVQ normaliza o espaço de entrada entre os valores [-0.8,0.8] e, portanto, o valor de g, neste caso, é igual a 1.6.

2.6 Algoritmo FCM

A teoria de conjuntos nebulosos diferencia-se da teoria clássica de conjuntos por permitir a utilização de pertinência a um intervalo entre os limites 0 e 1. Em outras palavras, podemos trabalhar não apenas com os valores 0 e 1, mas com todos entre estes valores, ou seja, o intervalo $[0, 1]$.

Isto quer dizer que um elemento pode pertencer a um conjunto com um grau de pertinência entre 0 e 1 (equação 2.7).

$$A = \{y, u_A(y), u_A(y) \in [0, 1]\} \tag{2.7}$$

onde A é um conjunto nebuloso, y representa um elemento do conjunto A e $u_A(y)$ é o grau de pertinência do elemento y ao conjunto A.

Em seguida é apresentado o algoritmo FCM, que realiza um agrupamento de dados através de conjuntos nebulosos.

O Algoritmo FCM (do inglês *"Fuzzy C-Means"*) é um dos métodos de agrupamento de dados baseado em conjuntos nebulosos mais utilizados. Ele permite que um dado pertença a dois ou mais agrupamentos simultaneamente [19, 20].

A idéia deste algoritmo é ponderar a distância de um ponto ao centro de agrupamento através de um grau (valor) de pertinência. Seu critério de erro é definido como:

$$J(m, \mathbf{W}) = \sum_{t=1}^{n} \sum_{i=1}^{K} u_i(t)^m d(\mathbf{x}(t), \mathbf{w}_i)^2 \tag{2.8}$$

onde m é qualquer número maior que 1. É o parâmetro que regula a forma das funções de pertinência e $u_i(t) = \mu_{\mathcal{C}_i}(\mathbf{x}(t))$ é o valor de pertinência do registro $\mathbf{x}(t)$ ao agrupamento $\mathcal{C}_i$. Desta maneira, um registro pode pertencer a mais de um agrupamento, de acordo com seu valor de pertinência.

A pertinência de todos os pontos de um conjunto de dados agrupados, forma a matriz de partição:

$$\mathbf{U} = \begin{bmatrix} u_{1,1} & \cdots & u_{K,1} \\ \vdots & \ddots & \vdots \\ u_{1,n} & \cdots & u_{K,n} \end{bmatrix} \tag{2.9}$$

A matriz **U** possui as seguintes propriedades:

- A soma dos valores de pertinência de um registro a todas as classes é igual a 1, ou seja, a soma de todas as colunas de uma linha da matriz **U** é igual a 1:

$$\forall \mathbf{x}(t) \in T, \sum_{i=1,...,K} \mathbf{u}_i(t) = 1 \tag{2.10}$$

- Cada registro deve pertencer a pelo menos um agrupamento e nenhum agrupamento pode conter todos os registros, ou seja, a soma de todas as linhas da Matriz **U** deve ser maior que 0 e menor que n:

$$\forall_i \in \{1, ..., K\}, 0 < \sum_{t=1,...,n} \mathbf{u}_i(t) < n \tag{2.11}$$

O Algoritmo FCM realiza uma otimização iterativa utilizando os valores de centro dos agrupamentos para o calculo da matriz **U** e a nova matriz **U** para o cálculo de valores para os centros dos agrupamentos.

O algoritmo FCM pode ser resumido nos seguintes passos:

1. Inicialização: os registros do conjunto de dados de treinamento são normalizados (equação 2.3); o número K de agrupamentos desejados é definido, tal que $2 < K < n$; o parâmetro de forma de função de pertinência ($m > 1$) é definido e uma estimativa inicial para o centro de agrupamentos **W** é definido.

2. A matriz de partição inicial é calculada (equação 2.13).

3. Enquanto o critério de parada não for alcançado, repita:

4. calcular os novos valores de centro de agrupamento a partir da matriz de partição calculada na iteração anterior:

$$\mathbf{w}_i = \frac{\sum_{t=1,...,n} \mathbf{u}_i(t)^m \mathbf{x}(t)}{\sum_{t=1,...,n} \mathbf{u}_i(t)^m} \tag{2.12}$$

5. Atualizar a matriz de partição:

$$\mathbf{u}_i(t) = \frac{1}{\sum_{j=1,\ldots,m} \left(\frac{d(\mathbf{x}(t),\mathbf{w}_i)}{d(\mathbf{x}(t),\mathbf{w}_j)} \right)^{\frac{2}{(m-1)}}} \tag{2.13}$$

   se houver algum registro **x**(t) que coincida com um centro de agrupamento $\mathbf{w}_i$, isto é, a distância $d(\mathbf{x}(t), \mathbf{w}_i(k)) = 0$, o vetor de pertinências $\mathbf{u}_i(t)$ é a linha i da matriz identidade 1^K.

6. Critério de parada.

O critério de parada de treinamento do algoritmo pode ser calculado a partir da variação dos centros de agrupamento e da matriz de partição:

$$\varepsilon_1 = d(\mathbf{W}_{q+1}, \mathbf{W}_q) < \delta_1 \qquad\qquad \varepsilon_2 = d(\mathbf{U}_{q+1}, \mathbf{U}_q) < \delta_2 \tag{2.14}$$

onde q é o valor da iteração.

BEZDEK [20] apresenta um teorema sobre a convergência do algoritmo FCM a um mínimo local do critério de erro (equação2.8).

2.7 Validação de Cluster

O principal interesse no processo de agrupamento é revelar a organização de padrões em grupos "sensíveis", que permitem a descoberta de similaridades e diferenças e a geração de produção de inferências úteis sobre eles.

Métodos de Agrupamento de dados utilizam algoritmos de classificação não-supervisionada. Por esta razão, não existem classes pré-definidas e nem exemplos que mostrem que tipos de relações desejáveis são válidas acerca dos dados. Os métodos de agrupamento utilizados neste trabalho permitem que sejam realizados particionamentos de dados em um número de classes determinado pelo usuário.

Suponha que um conjunto de dados seja dividido, inicialmente em 3 agrupamentos. Em seguida em 4 e depois em 5 e assim sucessivamente até 15 agrupamentos. Como não há classes pré-existentes, como é possível identificar o número de agrupamentos no qual o conjunto de dados é melhor agrupado?

Quando se trabalha em um espaço de duas ou três dimensões, ou seja, quando o conjunto de dados possui dois ou três atributos, é possível identificar o melhor agrupamento gerando-se um gráfico que contenha o conjunto de dados, as divisões dos agrupamentos e seus respectivos centros. Mas a maioria dos conjuntos de dados possui mais de três dimensões, o que descarta esta possibilidade.

Um índice de validação de agrupamento de dados é utilizado para resolver este problema. Ele apresenta, através de um conjunto de métricas, em quantos agrupamentos um determinado conjunto de dados é melhor agrupado.

2.8 Índice PBM

O Índice PBM [9], foi o índice de validação de agrupamentos de dados utilizado neste trabalho para validar os agrupamentos gerados pelos algoritmos descritos no capítulo anterior.

Para uma melhor compreensão de como o índice PBM é calculado, são apresentadas as seguintes notações:

- $\mathbf{U}(\mathbf{X})$ representa uma matriz de partição;
- $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ representa um conjunto de dados formado por n registros;
- K é o número de agrupamentos de dados.

A matriz de partição $\mathbf{U}(\mathbf{X})$ de tamanho $n \times K$, pode ser representada como $\mathbf{U} = [u_{ti}], t = 1, ..., n$ e $i = 1, ..., K$, onde u_{ti} é o valor de pertinência de $\mathbf{x}_t$, ao cluster $\mathcal{C}_i$.

Em particionamentos *"crisp"*, as seguintes condições ocorrem: $u_{ti} = 1$ se $\mathbf{x}_t \in \mathcal{C}_i$, caso contrário $u_{ti} = 0$. O propósito é classificar um conjunto de dados tal que

$\mathcal{C}_i \neq \emptyset$ para $i = 1, ..., K$,

$\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ para $i = 1, ..., K, j = 1, ..., K$ e $i \neq j$

No caso de agrupamento nebulosos (seção 2.6), o propósito é envolver uma matriz de partição apropriada $\mathbf{U} = [u_{ti}]_{n \times K}$, onde $u_{ti} \in [0, 1]$, tal que u_{ti} denote a grade de pertinência do t-ésimo elemento ao i-ésimo agrupamento. Em particionamentos nebulosos, as seguintes condições ocorrem:

$0 < \sum_{t=1}^{n} u_{ti} < n$ para $i = 1, ...K$,

$\sum_{i=1}^{K} u_{ti} = 1$ para $t = 1, ..., n$, conseqüentemente

$\sum_{i=1}^{K} \sum_{t=1}^{n} u_{ti} = n$.

O índice PBM é definido como:

$$PBM(K) = \left(\frac{1}{K} \times \frac{E_1}{E_K} \times D_K \right)^2, \tag{2.15}$$

onde K é o numero de agrupamentos e

$$E_K = \sum_{i=1}^{K} E_i, \tag{2.16}$$

tal que

$$E_i = \sum_{t=1}^{n} u_{ti} d(\mathbf{x}_t, \mathbf{w}_i). \tag{2.17}$$

e

$$D_K = \max_{i,j=1}^{K} d(\mathbf{w}_i, \mathbf{w}_j). \tag{2.18}$$

n é o número total de pontos (registros) em um conjunto de dados, $\mathbf{U}(\mathbf{X}) = [u_{ti}]_{n \times K}$ é uma matriz de partição para os dados e $\mathbf{w}_i$ é o centro do i-ésimo agrupamento. O objetivo é maximizar este índice afim de se obter o número real de agrupamentos, ou seja, o valor máximo deste índice indica o melhor particionamento.

Como apresentado na equação 2.15, o índice PBM é uma composição de três fatores: $1/K$, E_1/E_K e D_K. O primeiro fator diminui à medida em que o valor de K aumenta e isto, então, reduz o valor do índice. O segundo fator consiste da razão de E_1 (ponto central

do conjunto de dados), a qual é constante para um determinado conjunto de dados, e E_K, o qual diminui à medida em que o valor de K aumenta. Então, o valor do índice PBM aumenta à medida em que o valor de E_K diminui. Isto, por outro lado, indica que deveria ser encorajada a formação de maior número de agrupamentos, os quais são naturalmente compactados. Finalmente, o terceiro fator, D_K medindo o máximo de separação entre um par de agrupamentos, aumenta com o valor de K. Note que este valor é limitado pelo máximo de separação entre dois pontos no conjunto de dados [9].

Este capítulo apresentou detalhes sobre os métodos de agrupamentos de dados utilizados na fase de aplicação deste trabalho e uma descrição do índice de validação de agrupamento conhecido como PBM. No próximo capítulo são apresentados detalhes sobre a base de dados e os programas que utilizam os métodos de agrupamentos descritos neste capítulo.

Capítulo 3

Aplicação

3.1 A Base de Dados

A Base de dados utilizada na aplicação contém uma linha sísmica, no formato SEGY e a um registro de poço (*"log"*), cujo arquivo encontra-se no formato conhecido como **LAS**. Os dados foram cedidos à Universidade Federal do Rio de Janeiro pela ANP (Agência Nacional de Petróleo) e originam-se da Bacia de Camamu, localizada no litorial do Estado da Bahia. As coordenadas referentes à localização do poço cruzam com as coordenas da linha sísmica. Em outras palavras, o poço foi perfurado em uma área que pode ser visualizada na linha sísmica.

A ANP liberou mais quatro linhas sísmicas e mais dois registros de poços, mas estes não foram utilizados neste trabalho porque não havia registros de poços suficientes, ao longo de uma mesma linha sísmica, que permitisse utilizar os dois agrupamentos de dados integrados. Dados provenientes da linha sísmica "0247-5619" foram utilizados em todas as fases da aplicação enquanto o registro do poço "1BAS0118"foi utilizado por um especialista na identificação de pacotes de areias. Esta identificação foi possível graças ao perfil de raios gama presente no registro do poço, que possibilitou a localização destes pacotes na sísmica. A figura 3.1 exibe a localização da Bacia de Camamu.

FIGURA 3.1: Localização da Bacia de Camamu

3.2 Programas Utilizados

A figura 3.2 exibe um esquema gráfico correspondendo a uma visão macro da aplicação deste trabalho.

Subconjuntos de dados correspondentes à linha sísmica "0247-5619" são gerados como conjuntos de treinamento e aplicação. Os programas OpendTect, Matlab e SOM_PAK utilizam os métodos de agrupamentos. Em seguida os resultados produzidos por estes métodos são aplicados ao índice de validação de agrupamentos PBM no programa Matlab. Os resultados validados pelo índice PBM são visualizados em forma de janela sísmica no programa OpendTect.

3.2.1 OpendTect

O OpendTect, versão 1.2, é um programa da dGB (*http://www.dgb.nl*). Ele foi utilizado nas seguintes etapas da aplicação:

1. Importação e visualização da linha sísmica.

2. Geração e visualização de Atributos Sísmicos.

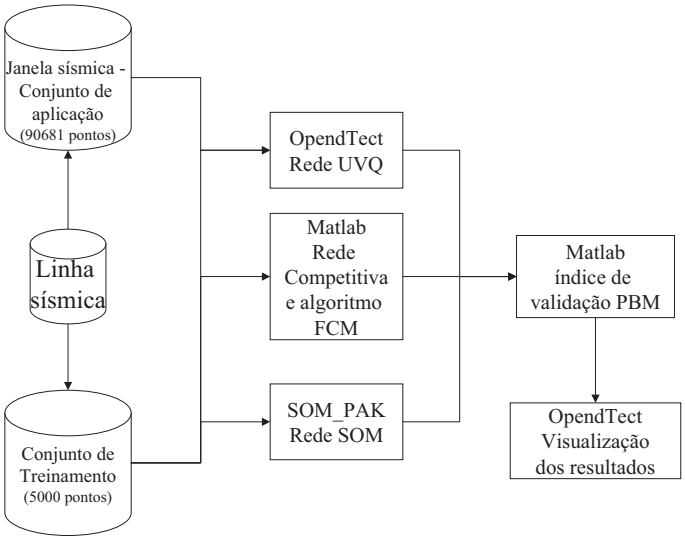


FIGURA 3.2: Visão Macro da Aplicação

3. Geração e visualização de redes neurais (rede UVQ).

4. Exportação das entradas e dos resultados de configurações de redes neurais.

5. Importação e visualização dos resultados obtidos pelos algoritmos de agrupamentos de dados utilizados no programa Matlab.

Importação e visualização da linha sísmica

Apesar do programa OpendTect só trabalhar com volumes sísmicos (sísmica 3D), é possível importar uma linha sísmica 2D simulando-se que esta faça parte de um volume 3D. Uma linha sísmica possui coordenadas X e Y, assim como um poço. Cada traço sísmico que compõe a linha, possui um par dessas coordenadas. Já um volume sísmico 3D, utiliza, além das coordenadas X e Y, coordenadas conhecidas como Inline e Crossline, já que um cubo é formado por várias linhas. Quando se deseja importar um arquivo no formato **SEGY**, o OpendTect permite ao usuário, informar os números dos bytes nos quais localizam-se as coordenadas Inline e Crossline. Em outras palavras, é possível utilizar outros valores para estas coordenadas, criando-se uma espécie de "pseudo-cubo". No caso da linha "0247-5619", foi informado ao programa que as coordenadas Inline e Crossline se encontravam, respectivamente, nos bytes **117** e **5**. O byte 5 refere-se a um valor que representa o traço sísmico atual, ou seja, varia de 1 a 1386. O byte 117 refere-se a uma unidade chamada **dt**,

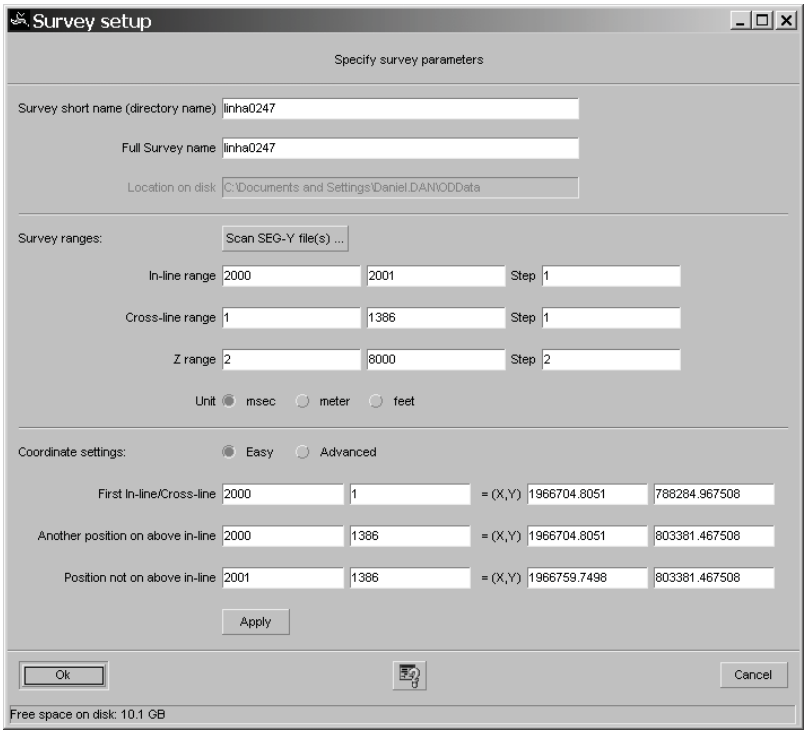


FIGURA 3.3: Configuração de Survey

cujo significado é intervalo de amostragem. Esta unidade possui o valor 2000 em todos os traços, ou seja, este valor permanece constante ao longo dos 1386 traços sísmicos.

Após a escolha dos bytes que substituirão as coordenadas Inline e Crossline, o OpendTect fornece uma ferramenta que converte medidas In-line e Crossline em coordenadas X e Y. Como o programa, no momento da criação de um *survey*, exige que estes dados sejam inseridos, esta ferramenta foi utilizada para converter os valores indicados como coordenadas Inline e Crossline em coordenadas X e Y.

Após ter todas as coordenadas solicitadas em mãos, um novo *survey* é criado. A figura 3.3 exibe a janela de configuração de um novo *survey* com as devidas coordenadas solicitadas preenchidas. Repare que o valor inicial da Coordenada Inline, em **Inline range** é 2000 enquanto o valor final é 2001. Não é possível criar um *survey* onde o valor inicial da coordenada Inline seja o mesmo que o valor final, mas é possível após a criação do *survey*, ignorar o valor 2001 e trabalhar apenas com o valor 2000.

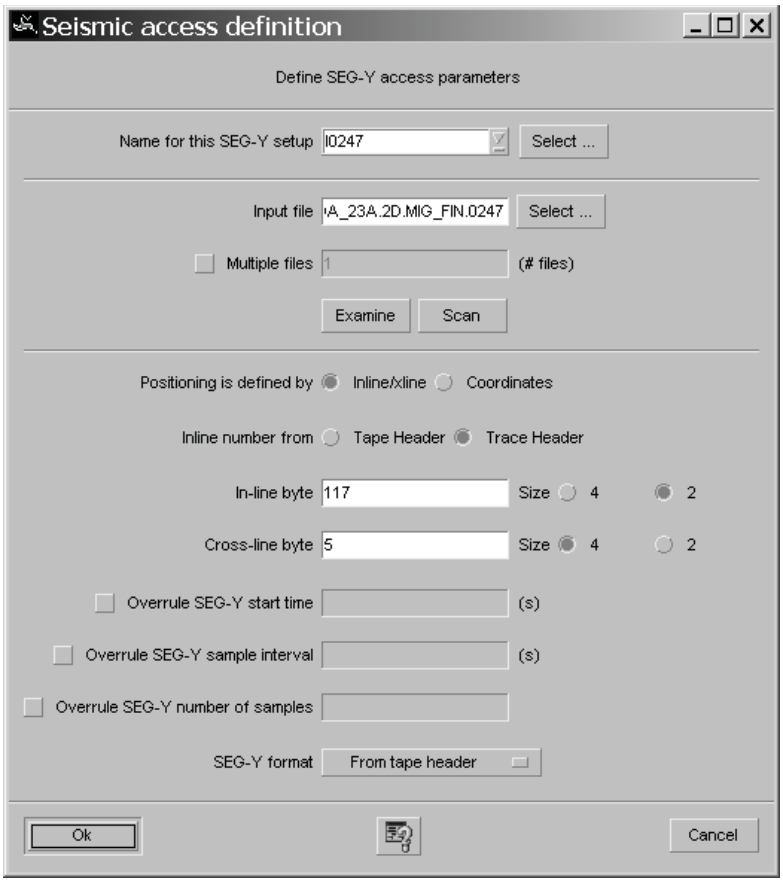


FIGURA 3.4: Configuração para a importação da linha sísmica "0247-5619"

Após a criação do *survey*, a linha sísmica é então importada, configurando-se a linha sísmica como descrito acima, como pode ser observado na figura 3.4.

Após a execução dos passos acima, a linha sísmica pode ser carregada e visualizada (Figura 3.5).

Geração e visualização de Atributos Sísmicos

É possível gerar diversos atributos sísmicos no programa OpendTect. São chamados atributos instantâneos aqueles calculados amostra por amostra. Os demais são calculados utilizando-se valores de mais de uma amostra, como energia e freqüência não-instantânea.

O programa OpendTect permite que sejam criados diferentes grupos de atributos e que os mesmos sejam salvos para posterior utilização. Após a criação de um grupo de atributos os mesmos podem ser visualizados no OpendTect. Mas este é um processo lento, pois o

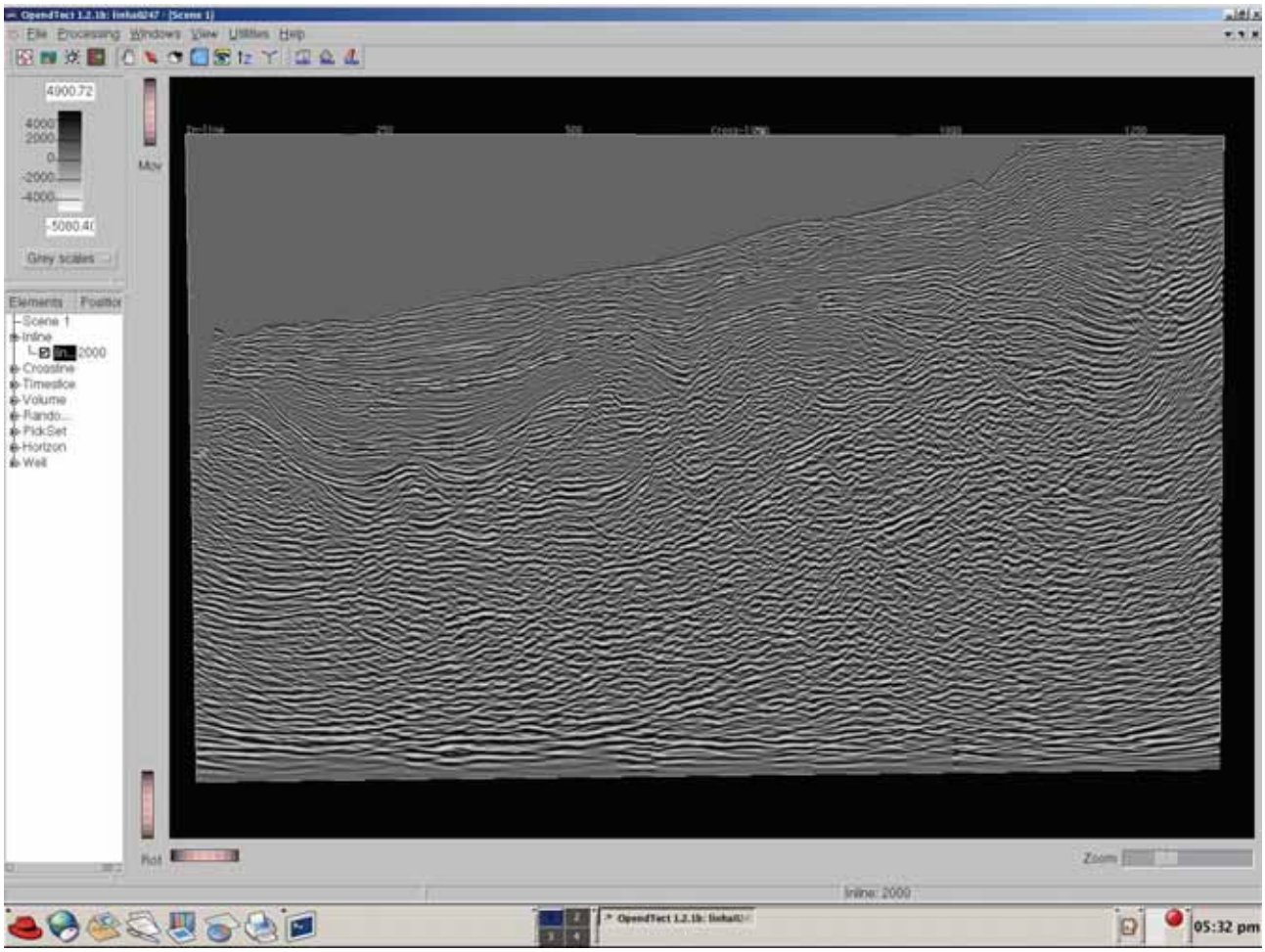


FIGURA 3.5: Visualização da linha "0247-5619" no Programa OpendTect

programa, antes de visualizar o atributo, calcula-o, amostra por amostra e traço por traço. Alguns atributos demoram vários minutos para serem calculados e visualizados. Há uma opção que resolve este problema. É possível criar volumes de atributos sísmicos. O usuário pode escolher um grupo de atributos sísmicos e selecionando um de cada vez, gerar um volume no formato interno do OpendTect cuja extensão de arquivo é **cbvs**. A criação destes volumes é lenta, mas após sua realização, é possível carregar um atributo sísmico em poucos segundos, melhorando o desempenho do carregamento dos atributos em aplicações futuras.

Outra vantagem de se criar um volume é a possibilidade de exportar os atributos sísmicos no formato **SEGY**, permitindo-se a importação do mesmo no programa Matlab.

Geração e visualização de redes neurais

O OpendTect possui uma rede neural que utiliza um algoritmo de agrupamento de dados UVQ. Como foi descrito na seção 2.5, a rede apresenta duas saídas para cada vetor de entrada: uma apresenta o número do cluster no qual o vetor de entrada foi agrupado e a outra apresenta o grau de competição (*"Match degree"*), que é uma espécie de grau de confiança, variando de zero a um, sendo que a proximidade ao valor um significa uma proximidade do vetor de entrada ao seu centro de classe.

Os vetores de entradas a serem utilizadas pela rede neural na fase de treinamento, são obtidas através de grupos de pontos *"picksets"*. Um grupo de pontos pode ser escolhido pelo usuário ou pode ser gerado aleatoriamente. Foram gerados três grupos de pontos. Dois deles apenas para identificar as áreas que possuem areias, sendo os pontos marcados pelo usuário. O terceiro foi gerado aleatoriamente dentro de uma faixa de valores próximo às areias conhecidas, ou seja, próximas ao poço "1BAS0118BA". Este grupo possui cinco mil pontos e será utilizado para o treinamento dos métodos de agrupamentos de dados. O objetivo de escolher uma área próxima aos pacotes de areias é recolher uma quantidade significativa de pontos em uma área conhecida, buscando disponibilizar aos algoritmos de agrupamento de dados, dados que representem os pacotes de areias e dados que representem quaisquer outras litologias.

No OpendTect, as áreas em torno dos pacotes de areia são marcadas utilizando-se grupos de pontos, como pode ser observado na figura 3.6.

Um conjunto de dados (grupo de pontos) similar aos que serão utilizados como dados de entrada para o treinamento dos algoritmos de agrupamento de dados pode ser visualizado na figura 3.7.

Uma rede neural utiliza um grupo de pontos da seguinte forma: Para cada ponto do grupo, são recolhidos os valores dos atributos sísmicos que foram escolhidos para fazerem parte da entrada dos vetores de entrada da rede neural. Por exemplo: Se os atributos amplitude, freqüência instantânea, energia e fase foram escolhidos para fazerem parte da entrada de uma rede, cada ponto deverá possuir valores destes quatro atributos. O resultado da união entre um grupo de pontos e um grupo de atributos sísmicos é uma matriz bidimensional. Esta matriz contém um número de registros igual ao número do grupo de pontos gerados no OpendTect, e um número de colunas (atributos) igual ao número de atributos sísmicos selecionados.

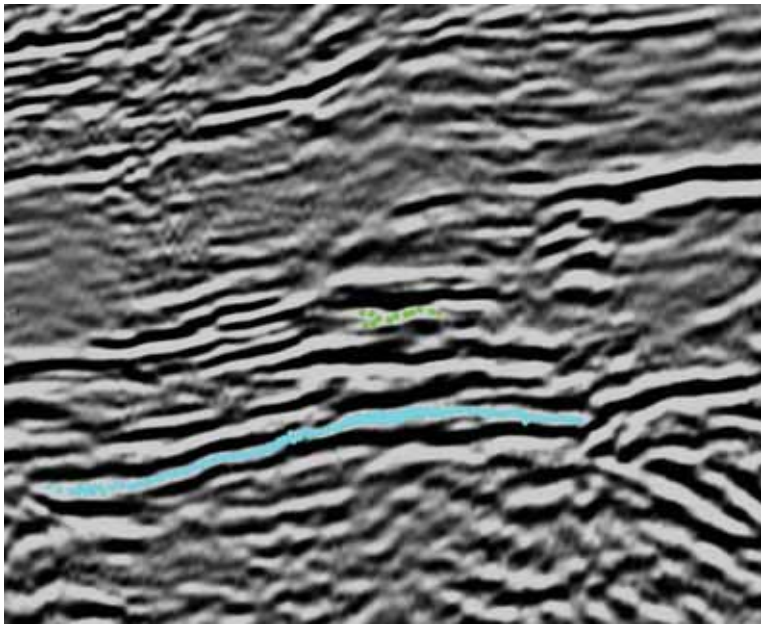


FIGURA 3.6: Grupos de pontos (em verde e azul) exibindo áreas em torno dos pacotes de areia identificados na sísmica gerados no programa OpendTect.

Suponha que seja gerado um grupo de cinco mil pontos aleatórios em torno de uma linha sísmica e que, utilizando-se a mesma linha sísmica, sejam gerados três atributos sísmicos diferentes. O conjunto de dados utilizados na fase de treinamento de uma rede neural será composto por uma matriz bidimensional contendo 5000 pontos e três atributos por ponto.

A figura 3.8 exibe a janela de configuração de redes neurais no OpendTect.

É possível escolher quais atributos serão utilizados como entrada na rede, qual conjunto de pontos será utilizado no treinamento da rede, em quantos agrupamentos deseja-se agrupar os dados de entrada e a opção não-supervisionada (*"unsupervised"*) deverá ser marcada para a utilização da rede UVQ.

Uma configuração foi criada, utilizando-se quatro atributos: cosseno da fase, energia, freqüência média e amplitude. O atributo cosseno da fase exibe variações na fase sobre uma escala rotativa e provê informações estruturais. O atributo amplitude destaca regiões de interesse estratigráfico e detecta regiões com baixa ou alta amplitude, em outras palavras, ele mede a intensidade da refletividade e está associada à energia do sinal sísmico. O atributo fase é uma medida que permite visualizar a continuidade de um evento e a razão de mudança no tempo do atributo fase é expresso pelo atributo freqüência.

O resultado visual do treinamento e de seu resultado pode ser observado na figura 3.9.

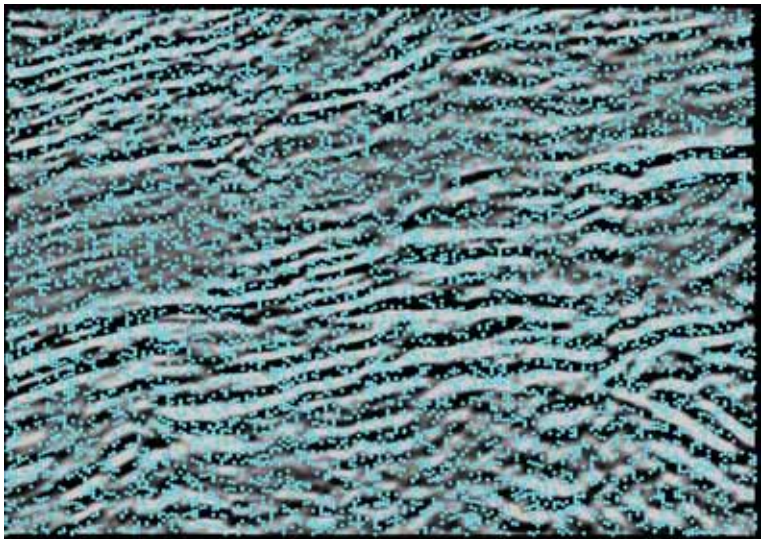


FIGURA 3.7: Grupo de Pontos que serão utilizados no treinamento da Rede UVQ do Opend-Tect (em azul).

Outras configurações foram criadas, contendo a mesma base de teste e os mesmos atributos de entrada, porém, o valor de agrupamentos (classes), foi sendo incrementado. No total, onze configurações foram criadas, sendo que a única diferença entre elas foi o número de agrupamentos, que variou de 5 a 15.

Exportação das entradas e dos resultados de configurações de redes neurais

Os resultados das diferentes configurações de redes neurais gerados na etapa acima, foram convertidos no formato **cbvs** e exportados no formato **SEGY** para serem, futuramente, comparados com os resultados dos outros algoritmos de agrupamento de dados. De maneira semelhante, o grupo de pontos utilizado no treinamento das redes foram copiados e preparado para também ser utilizado pelos demais algoritmos de agrupamento. Cada um dos quatro atributos: amplitude, energia, média da freqüência e cosseno da fase foram exportados no formato **SEGY**. Estes passos foram executados a fim de utilizar o mesmo conjunto de dados nos diferentes algoritmos de agrupamentos de dados.

Importação e visualização dos resultados obtidos pelos algoritmos de agrupamentos de dados utilizados no programa Matlab

A Aplicação dos demais algoritmos de agrupamentos de dados foi realizada no programa Matlab. Apesar de ser possível gerar figuras e visualizar os resultados dos algoritmos de

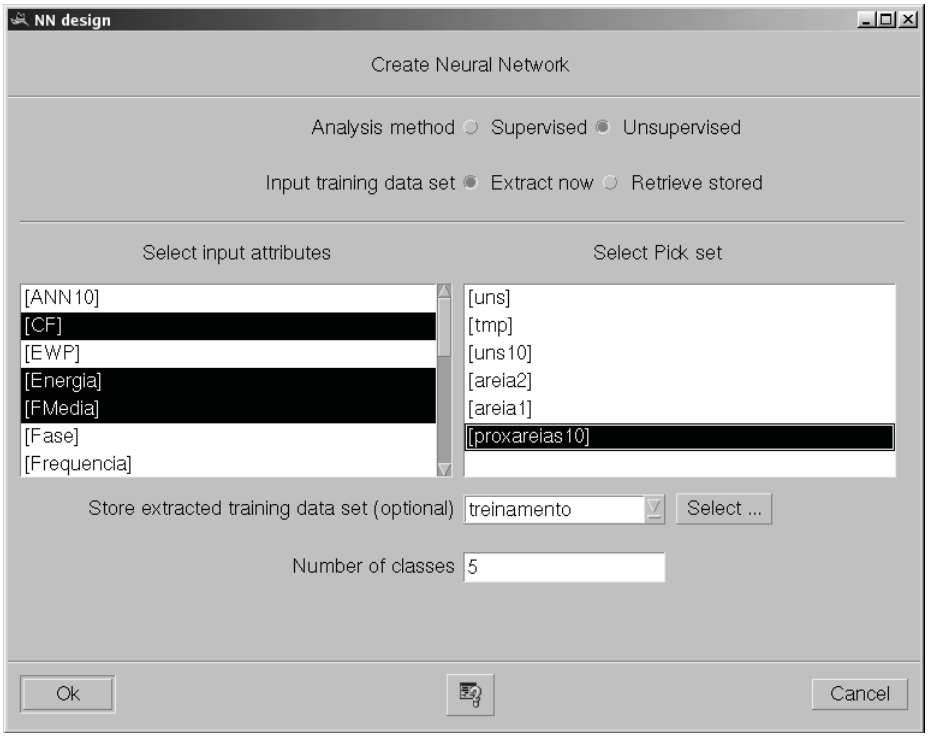
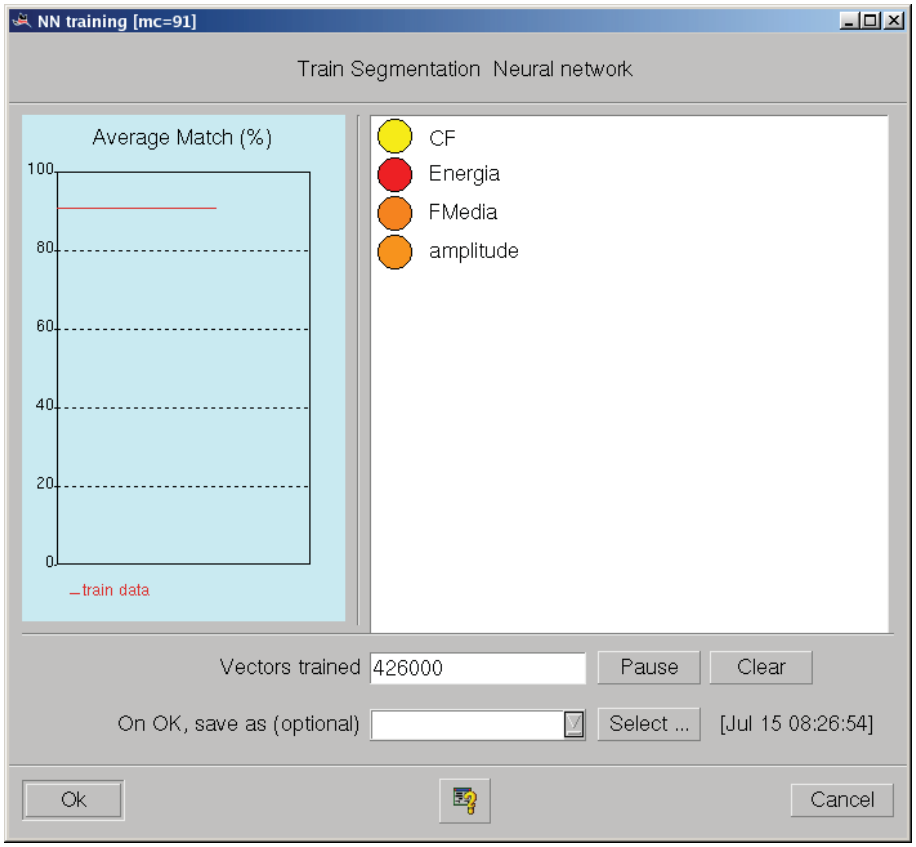
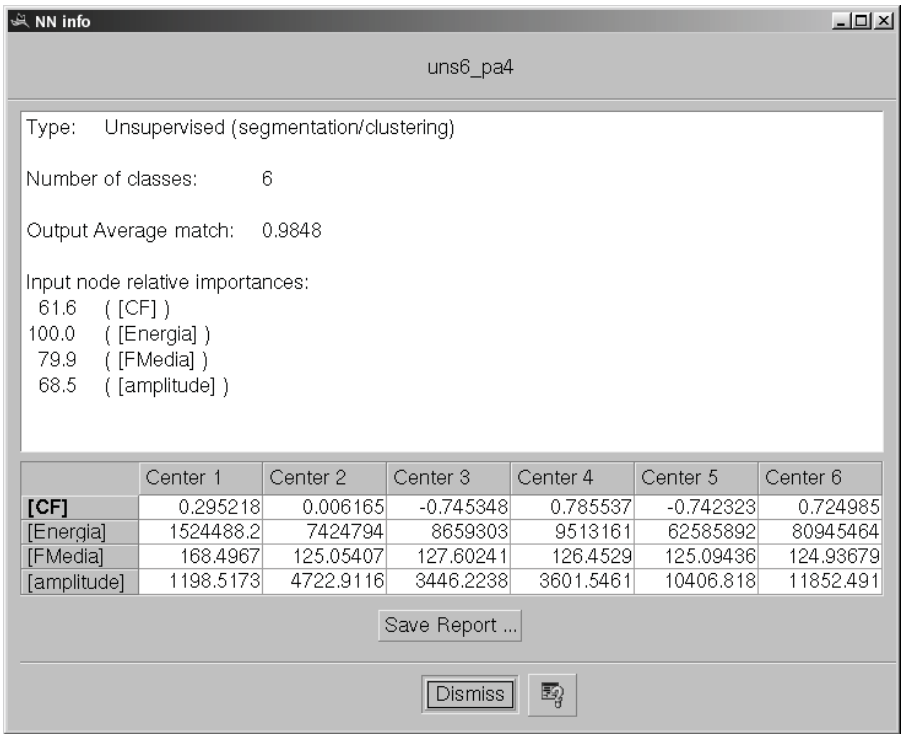


FIGURA 3.8: Janela de Configuração de Rede Neural no OpendTect.

agrupamento no Matlab, o OpendTect possui uma resolução muito superior e, portanto, os melhores resultados destes algoritmos serão importados e visualizados pelo OpendTect.

Estas foram as tarefas executadas pelo programa OpendTect. Em seguida, serão descritas as tarefas executadas no programa Matlab.





	Center 1	Center 2	Center 3	Center 4	Center 5	Center 6
[CF]	0.295218	0.006165	-0.745348	0.785537	-0.742323	0.724985
[Energia]	1524488.2	7424794	8659303	9513161	62585892	80945464
[FMedia]	168.4967	125.05407	127.60241	126.4529	125.09436	124.93679
[amplitude]	1198.5173	4722.9116	3446.2238	3601.5461	10406.818	11852.491

FIGURA 3.9: Treinamento e Relatório Gráfico do Programa OpendTect

3.2.2 Matlab e SOM_PAK

O Matlab é uma linguagem de alto nível desenvolvida pela MathWorks, objetivando atender necessidades de engenheiros e matemáticos. Mas ele, em sua versão 6.5 possui ferramentas (*"toolboxes"*) que ampliam as opções do programa. Como exemplo, cita-se a ferramenta de redes neurais utilizadas neste trabalho. O programa possui dentre vários algoritmos de redes neurais, o modelo de rede competitiva (2.3). Apesar de também ter a rede conhecida por SOM (*"Self Organizing Maps"*), Mapas Auto-Organizáveis (2.4), esta não foi utilizada no Matlab e sim no programa SOM_PAK. Este programa permite aplicar a rede SOM com um desempenho muito superior à caixa de ferramentas do Matlab e, por isso, foi utilizado. Este programa foi desenvolvido pela Universidade de Helsinque e o próprio Kohonen, criador desta rede, participou de sua implementação [16].

Além da utilização das duas redes, foram implementados diversas rotinas e funções para importar, preparar, gerar as redes e exportar os dados. Além disso um último algoritmo de agrupamento de dados, baseado em conjuntos nebulosos (*"fuzzy"*), conhecido como FCM foi implementado na linguagem do Matlab. Uma função de validação de cluster, conhecida como índice PBM, foi também implementada para validar e comparar os resultados de todos os algoritmos de agrupamento. A figura 3.10, exibe um esquema gráfico contendo uma visão macro das atividades executadas no Matlab.

Em seguida são descritas as principais tarefas executadas pelas rotinas e funções utilizadas neste trabalho.

3.2.2.1 Importa_janela

Esta rotina é responsável por importar dados de atributos sísmicos referentes a uma janela sísmica e salvá-los no formato binário do matlab. Abaixo segue uma definição de janela sísmica e informações sobre como esta importação é realizada.

Uma janela sísmica, neste trabalho, é definida como um conjunto de dados igual ou menor que uma linha sísmica. Em outras palavras, por questões relacionadas ao desempenho e ao processamento de uma área específica em uma linha sísmica, é possível utilizar uma área em torno de um traço inicial e final e um tempo (amostra) inicial e final da linha sísmica. Este trabalho utiliza um conjunto de dados que se refere a uma janela sísmica em uma área próxima aos pacotes de areia localizados. Após a fase de treinamento, este conjunto de dados

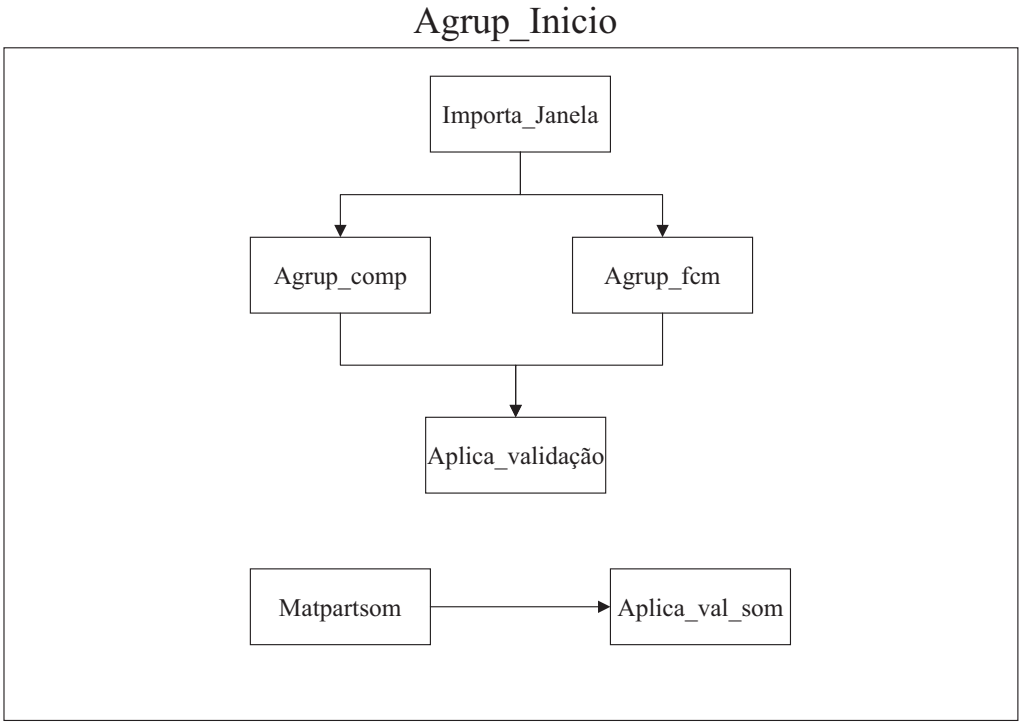


FIGURA 3.10: Esquema Gráfico representando as principais atividades realizadas no Matlab.

é aplicado nos algoritmos de agrupamentos. Por ora, suponha que todos os traços e amostras serão utilizados no processamento.

É necessário importar arquivos no formato **SEGY** (exportados pelo OpendTect). Em outras palavras, os mesmos dados que foram aplicados em uma rede UVQ do programa OpendTect, precisam ser importados no programa Matlab para serem aplicados pelos demais algoritmos de agrupamentos de dados.

O Matlab, em sua versão 6.5, não possui nenhum recurso que permita importar arquivos no formato **SEGY**, mas através de um conjunto de rotinas e funções desenvolvidas para este programa, conhecidas como **SegyMAT**, é possível importar dados neste formato. Elas podem ser encontradas em *"http://segymat.sourceforge.net/"*, juntamente com sua documentação.

Através do SegyMAT é possível importar os atributos sísmicos gerados pelo OpendTect e separar os dados sísmicos dos cabeçalhos existentes no formato **SEGY**, permitindo-se

que estes dados sejam salvos no formato binário do Matlab (**"mat"**), facilitando futuros processamentos.

Após a importação dos dados é necessário prepará-los para que os mesmos possam ser utilizados como entrada nos algoritmos de agrupamento de dados. Para compreender o formato desta entrada, é importante lembrar o formato da matriz bidimensional utilizada pelo OpendTect para treinar a rede neural UVQ (seção 3.2.1). Esta matriz pode ser exportada pelo OpendTect e importada pelo Matlab sem muito esforço. Deve-se apenas fazer:

- uma cópia do arquivo, que é exportado no formato **ASCII**;
- verificar a posição dos atributos sísmicos (em quais colunas estão os atributos e em que ordem);
- apagar o texto referente ao cabeçalho deste arquivo e
- excluir as colunas que fazem referência à localização dos pontos

Após o treinamento por um algoritmo de agrupamento de dados, os dados restantes, referentes aos quatro atributos da janela sísmica exportada pelo OpendTect, são aplicados nos algoritmos de agrupamento de dados.

O problema é que estes dados não são importados em um formato que permita utilizá-los como dados de entrada de uma rede neural ou um algoritmo FCM. Estes dados devem sofrer uma transformação para que possam ser utilizados por estes algoritmos. A figura 3.11 explica como esta transformação é realizada, supondo que todos os traços e amostras da linha sísmica "0247-5619" fossem utilizados no conjunto de dados de aplicação.

Ao lado esquerdo observa-se as matrizes dos atributos sísmicos que são carregadas pela rotina. Cada círculo colorido representa o primeiro elemento de cada matriz, ou seja, a primeira amostra do primeiro traço sísmico. Todos os círculos passam a fazer parte do primeiro registro da matriz bidimensional que se encontra à direita na figura. Esta matriz armazenará cada elemento das matrizes de atributos neste sentido. Em outras palavras, a segunda amostra do primeiro traço das matrizes de cada atributo formará o segundo registro da matriz à direita. O registro de número 3997 da matriz à direita, será formado pela primeira amostra do segundo traço sísmico das matrizes de cada atributo à esquerda, e assim em diante.

Cada atributo possui 3996 amostras e não 4000 amostras por traço como na linha sísmica. Isto porque os atributos não-instantâneos freqüência média e energia, foram calculados a

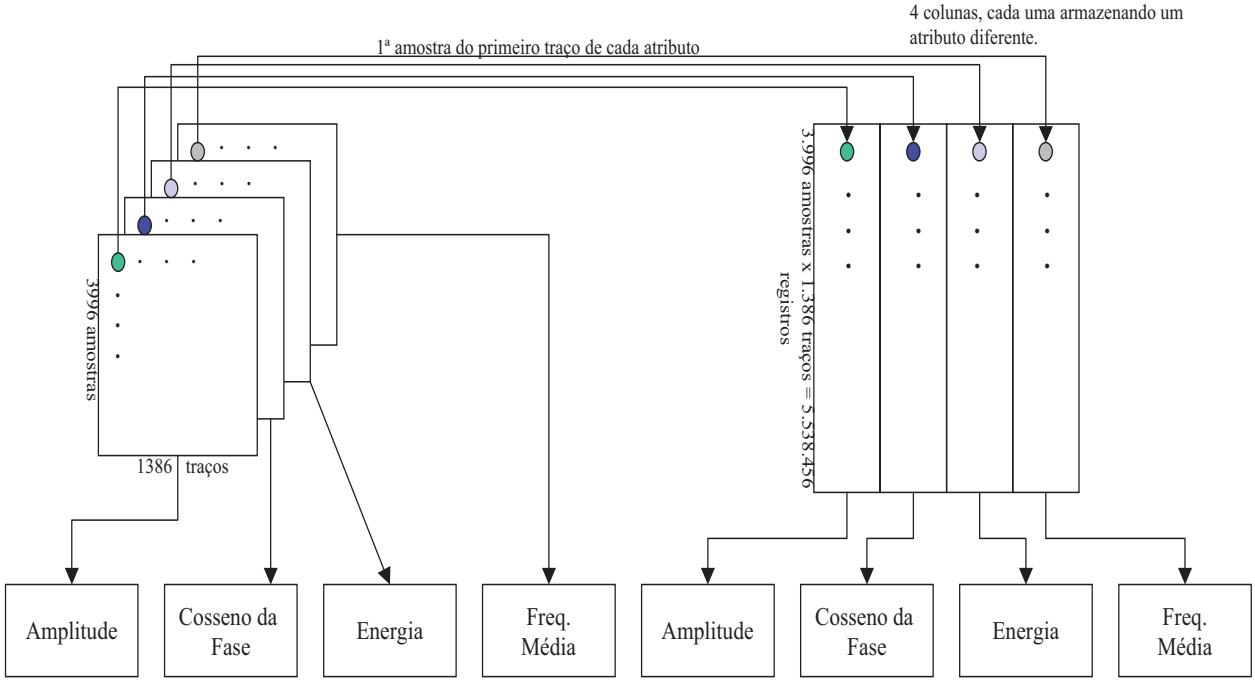


FIGURA 3.11: Transformação de matrizes de atributos sísmicos.

cada quatro milisegundos. Como a primeira amostra inicia-se em 2 ms e termina em 8 segundos e há um espaço de 2ms de tempo entre cada amostra, ao calcular estes atributos, os mesmos reduziram o tempo em um total de 8 ms (os 4 ms iniciais e os 4 ms finais). Conseqüentemente, o número de amostras é reduzido em 4, pois as 2 primeiras e 2 últimas amostras são ignoradas.

Não é possível utilizar dados faltantes em algoritmos de agrupamento de dados e, portanto, cada atributo sísmico instantâneo teria suas duas primeiras e duas últimas amostras ignoradas, no momento em que fosse necessário exportá-la pelo programa OpendTect (seção 3.2.1).

Supondo que estes dados fossem utilizados como dados do conjunto de aplicação em um algoritmo de agrupamento de dados, o número de pontos (registros) da matriz bidimensional seriam iguais a $3.996 \times 1.386 = 5.538.546$.

3.2.2.2 Agrup_inicio

Esta é a rotina inicial das tarefas no Matlab. Através dele, as demais rotinas e funções são chamadas e algumas tarefas fundamentais são executadas. Abaixo, segue uma descrição

de suas tarefas:

- A rotina importa_janela (3.2.2.1) é chamada.
- Os conjuntos de dados que serão utilizados nas fases de treinamento e aplicação são carregados.
- O número de épocas a ser utilizado no treinamento das redes neurais é definido.
- Um número inicial e final de agrupamentos é definido (o número de agrupamentos inicial $= 5$ e o final $= 15$, ou seja, em cada um dos algoritmos de agrupamento de dados, serão geradas configurações com $5, 6, 7..., 15$ classes.
- São passados parâmetros a funções que serão responsáveis pelo treinamento e aplicação dos algoritmos competitivo e FCM. O treinamento da rede SOM é realizado no programa SOM_PAK. Os resultados produzidos (matrizes de partições) são salvos.
- Ao final, a rotina aplica_validação é chamada para que os resultados (matrizes de partições) sejam validados pela função índice PBM.

3.2.2.3 Agrup_comp e Agrup_fcm

Estas funções são responsáveis pelo treinamento e aplicação dos respectivos conjuntos de dados utilizando-se as redes neurais competitiva e o algoritmo FCM. Os resultados produzidos referem-se às matrizes de partição.

Todos os algoritmos de agrupamento de dados dividiram o conjunto de dados em 11 diferentes agrupamentos, como realizado na rede UVQ e descrito na seção 3.2.1. O particionamento inicial contém 5 agrupamentos e o final, 15.

O treinamento da rede SOM é realizado pelo programa SOM_PAK [16]. Para cada agrupamento da rede SOM, os neurônios da camada intermediária foram configurados como mostra a tabela (tabela 3.1). Após o treinamento da rede SOM, os pesos da rede são salvos no formato ASCII e posteriormente carregados no Matlab através da rotina **matpartsom**, que também realiza a aplicação do conjunto de dados de aplicação e salva as matrizes de partição no formato Matlab.

TABELA 3.1: Organização dos neurônios na camada intermediária - rede SOM.

Partição	Organização dos Neurônios na camada SOM
5	5×1
6	6×1
7	7×1
8	8×1
9	9×1
10	10×1
11	11×1
12	12×1
13	13×1
14	14×1
15	15×1

3.2.2.4 Aplica_validação e Aplica_val_som

Estas rotinas e algumas funções utilizadas pelas mesmas, são responsáveis por aplicar o índice de validação PBM aos resultados obtidos pelas funções Agrup_comp, Agrup_fcm, rede UVQ e pelos resultados produzidos pela rede SOM do programa SOM_PAK (matrizes de partição). Os resultados desta aplicação são salvos para futura análise e comparação.

Na seção 2.8, são descritos as expressões matemáticas implementadas nesta rotina.

Uma última rotina não exibida na figura 3.10, chamado **Gera_matpart_uvq** é utilizada para transformar os resultados de agrupamentos realizados pela rede UVQ em uma matriz bidimensional. Esta matriz é semelhante às matrizes geradas pelos algoritmos de agrupamento no Matlab e será utilizada para aplicar o índice PBM aos resultados obtidos pela rede UVQ no programa Matlab.

O próximo capítulo apresenta uma descrição do índice PBM e uma análise dos resultados produzidos por sua aplicação.

Detalhes sobre o programa Matlab são encontrados em [21, 22].

Capítulo 4

Resultados

Este capítulo apresenta os resultados da aplicação do índice aos agrupamentos de dados gerados pelos algoritmos de agrupamento de dados descritos no capítulo 3. Em seguida, é apresentada uma análise dos melhores resultados.

4.1 Apresentação dos Resultados

4.1.1 Detalhes sobre os conjuntos de dados

Nas seções 3.2.1 e 3.2.2.1 foram descritas etapas de exportação e organização dos conjuntos de dados utilizados nas etapas de treinamento e aplicação dos algoritmos de agrupamento de dados utilizados neste trabalho. Foi também descrito que o conjunto de dados de aplicação pode representar uma janela sísmica (3.2.2.1). Foi utilizado um conjunto de aplicação correspondente a uma janela sísmica cujos traços sísmicos e amostras envolvem uma área em torno dos pacotes de areias. Esta área corresponde a 501 traços e 181 amostras por traços e, portanto, o conjunto é formado por uma matriz contendo $501 \times 181 = 90681$ pontos e 4 atributos por ponto (cosseno da fase, amplitude, energia e freqüência média). Esta transformação ocorre segundo o processo que transforma uma janela sísmica em uma matriz bidimensional (seção 3.2.2.1 e figura 3.11).

O conjunto de dados de treinamento corresponde a 5 mil pontos e 4 atributos por pontos. Este conjunto foi formado por grupos de pontos aleatórios da janela sísmica referente ao primeiro conjunto de dados (figura 3.7). A figura 4.1 exibe detalhes sobre a localização desta janela na linha "0247-5619".

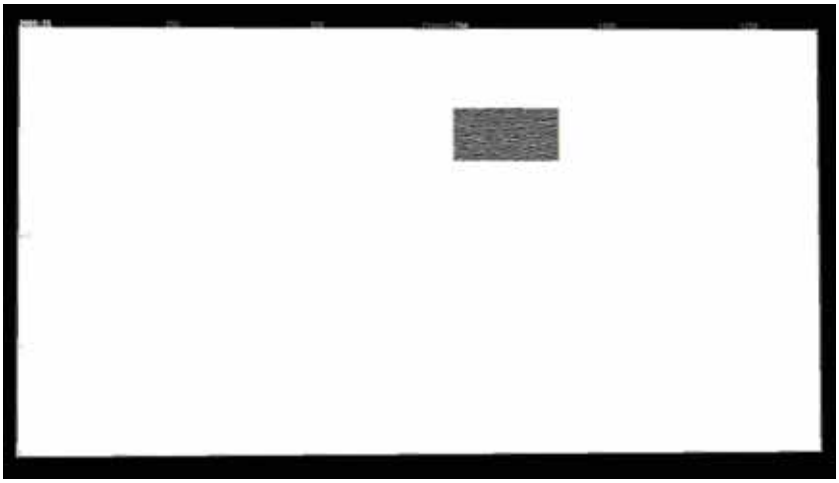


FIGURA 4.1: Janela sísmica referente ao primeiro conjunto de dados de aplicação.

As rotinas **aplica_validacao** e **aplica_val_som**, acompanhadas das funções **validacao**, e **validacaof** são responsáveis pela aplicação da validação dos agrupamentos (figura 4.2).

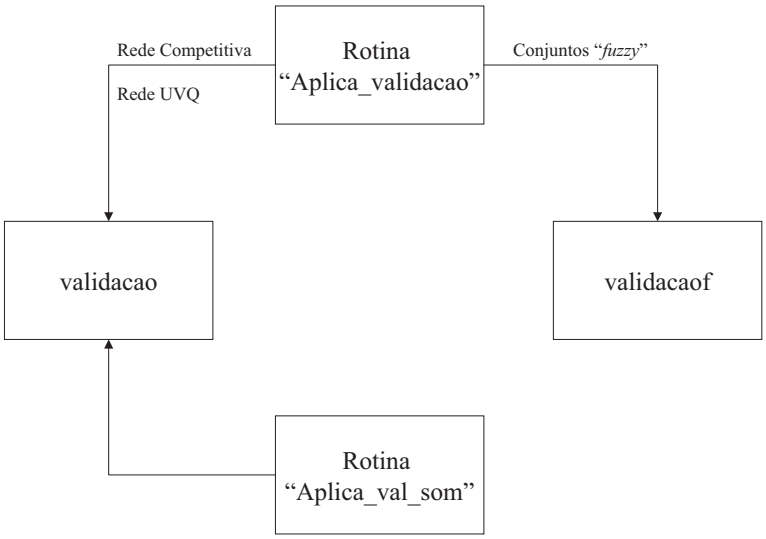


FIGURA 4.2: Rotinas responsáveis pela aplicação do índice de validação PBM.

4.1.2 Resultados

Abaixo seguem os resultados obtidos com a aplicação do índice PBM. Os melhores resultados de cada configuração ou algoritmo são destacados em negrito.

Inicialmente são apresentados os valores dos índices PBM aplicados às matrizes de partição do algoritmo FCM. Cada coluna da tabela 4.1 possui valores correspondentes à m (forma de pertinência) variando de 1.1 a 1.5.

Os resultados obtidos pelas redes neurais são apresentados na tabela 4.2.

TABELA 4.1: Resultados da aplicação do índice PBM aos agrupamentos do algoritmo FCM.

Classes	(m = 1.1)	(m = 1.2)	(m = 1.3)	(m = 1.4)	(m = 1.5)
5	3,13E+14	3,12E+14	3,02E+14	2,80E+14	2,49E+14
6	3,54E+14	3,48E+14	3,28E+14	2,95E+14	2,53E+14
7	2,90E+14	2,84E+14	2,67E+14	2,39E+14	2,10E+14
8	2,50E+14	2,43E+14	2,30E+14	2,08E+14	1,81E+14
9	6,23E+14	3,05E+14	4,90E+14	1,81E+14	**3,34E+14**
10	**6,65E+14**	**5,83E+14**	**5,31E+14**	**3,83E+14**	2,93E+14
11	6,59E+14	5,33E+14	4,75E+14	3,60E+14	2,55E+14
12	6,27E+14	5,56E+14	4,92E+14	3,77E+14	2,59E+14
13	5,69E+14	5,18E+14	4,48E+14	3,24E+14	2,36E+14
14	5,07E+14	4,20E+14	1,84E+14	3,05E+14	2,48E+14
15	4,78E+14	4,26E+14	3,55E+14	2,83E+14	2,12E+14

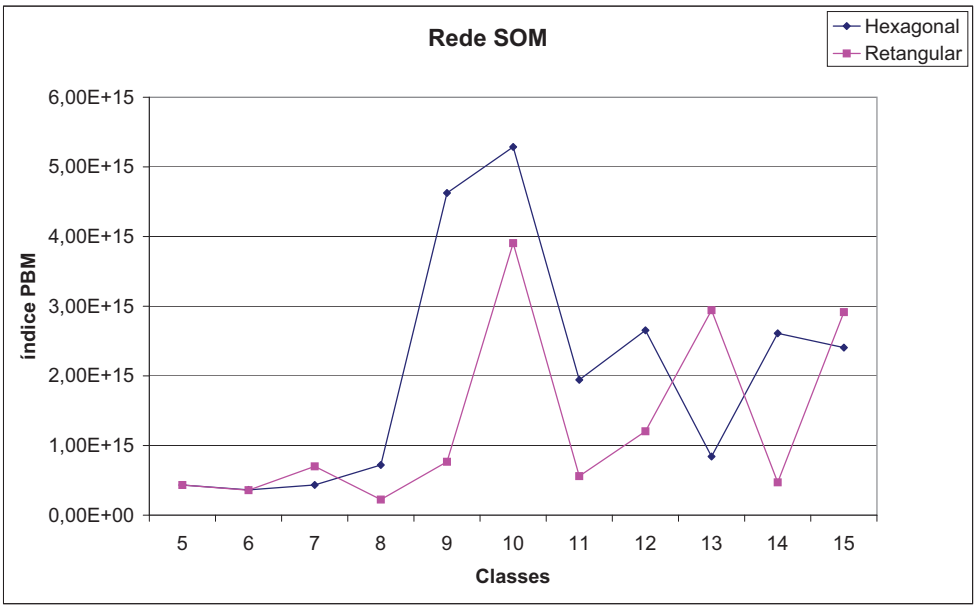
TABELA 4.2: Resultados da aplicação do índice PBM aos agrupamentos de Redes Neurais

Classes	SOM - Top. Hexagonal	SOM- Top. Retangular	Rede UVQ	Rede Comp.
5	4,33E+14	4,33E+14	2,94E+14	3,39E+14
6	3,63E+14	3,58E+14	1,30E+14	2,70E+14
7	4,34E+14	7,01E+14	3,19E+14	3,88E+14
8	7,21E+14	2,24E+14	**8,40E+14**	**7,16E+14**
9	4,63E+15	7,67E+14	4,26E+14	5,90E+14
10	**5,29E+15**	**3,90E+15**	4,07E+14	3,34E+14
11	1,94E+15	5,60E+14	3,85E+14	3,53E+14
12	2,65E+15	1,20E+15	3,17E+14	4,25E+14
13	8,42E+14	2,94E+15	2,89E+14	2,27E+14
14	2,61E+15	4,71E+14	2,19E+14	1,15E+14
15	2,41E+15	2,91E+15	2,25E+14	5,60E+13

As figuras que seguem exibem os resultados apresentados nas tabelas na forma de gráficos.

A figura 4.3 exibe gráficos dos índices obtidos pelas redes SOM e Competitiva.

A figura 4.4 exibe gráficos dos índices obtidos pelos resultados dos algoritmos da rede UVQ e FCM.



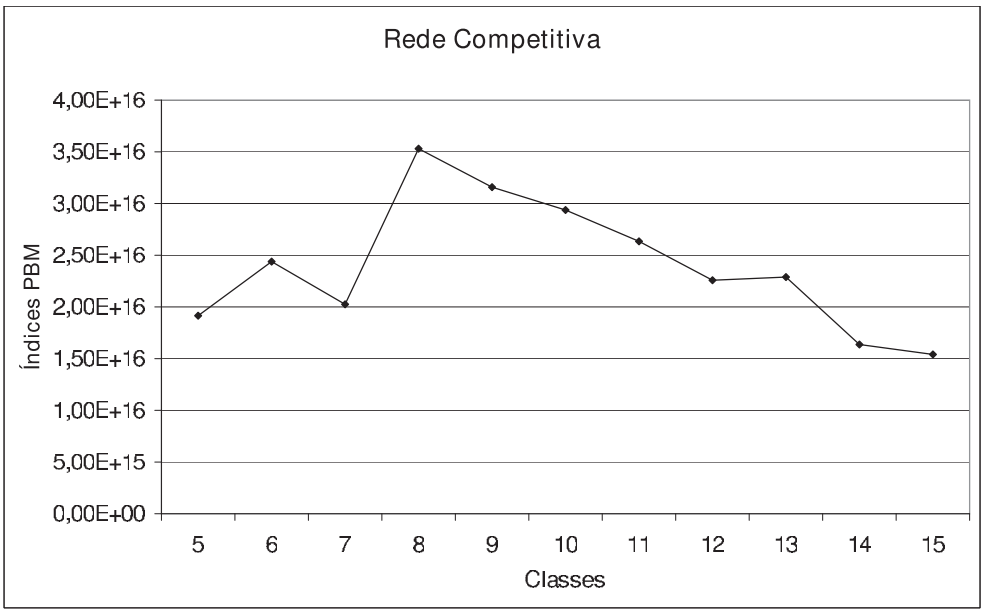
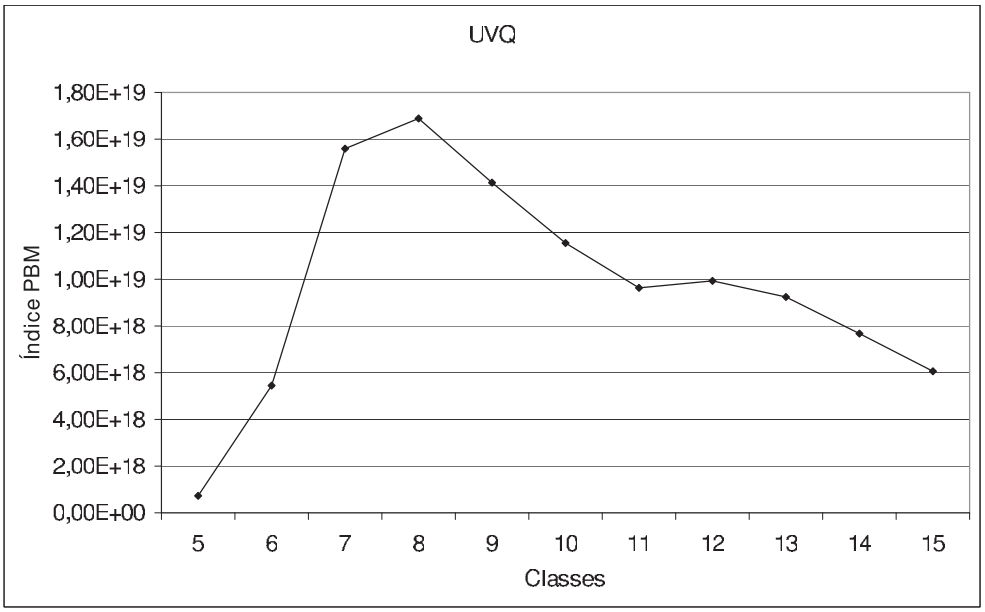


FIGURA 4.3: Resultados dos índices de validação de agrupamentos (Redes SOM e Competitiva).



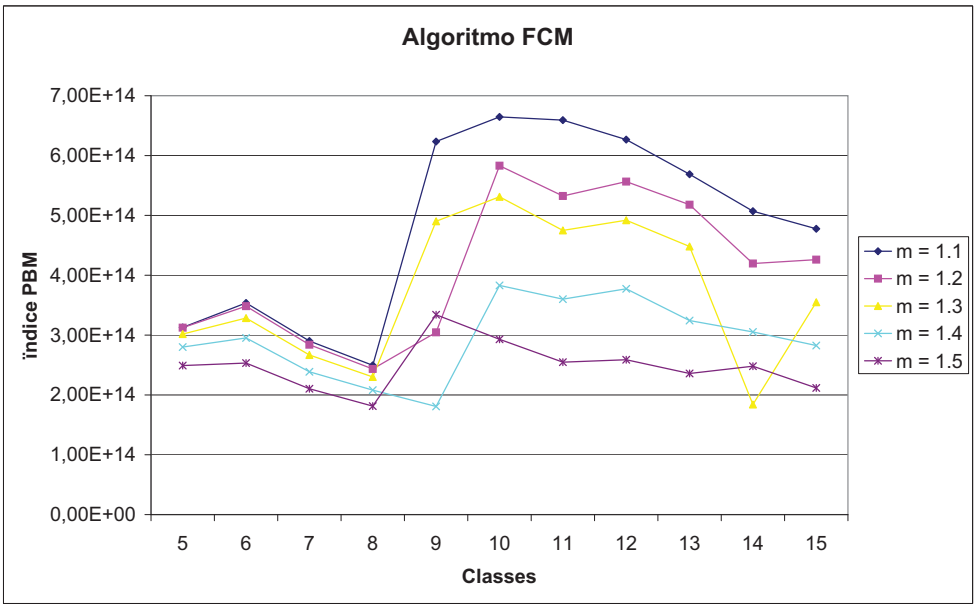


FIGURA 4.4: Resultados dos índices de validação de agrupamentos - rede UVQ e algoritmo FCM. No algoritmo FCM, o valor de m varia de 1.1 a 1.5, com passo de 0.1 (equação 2.8).

4.1.3 Análise dos Resultados

Como pode ser observado pelas tabelas e gráficos apresentados, os melhores resultados obtidos pela aplicação dos índices PBM às redes UVQ e competitiva apontaram para oito agrupamentos (classes), isto é, segundo o índice PBM, o conjunto de dados de aplicação é melhor agrupado em oito classes. Na rede SOM e no algoritmo FCM, os maiores índices apontam para dez classes, tendo a rede SOM apresentado o maior índice geral ($5,29E+15$), utilizando-se a topologia hexagonal.

Os resultados das redes UVQ e SOM são apresentados em seguida na forma de janelas sísmicas (figuras 4.5 e 4.6). Logo em seguida são apresentadas janelas contendo imagens dos atributos sísmicos utilizados neste trabalho (figuras 4.7, 4.8, 4.9 e 4.10). O resultado da rede e cada atributo sísmico é apresentado em duas figuras, sendo que na segunda são exibidos pontos em torno da localização dos pacotes de areia. Estas imagens mostram claramente que os agrupamentos formados pelas redes UVQ e SOM não são naturalmente detectáveis, ou seja, não são vistos claramente nos atributos sísmicos em separado.

No próximo capítulo é apresentada a conclusão deste trabalho.

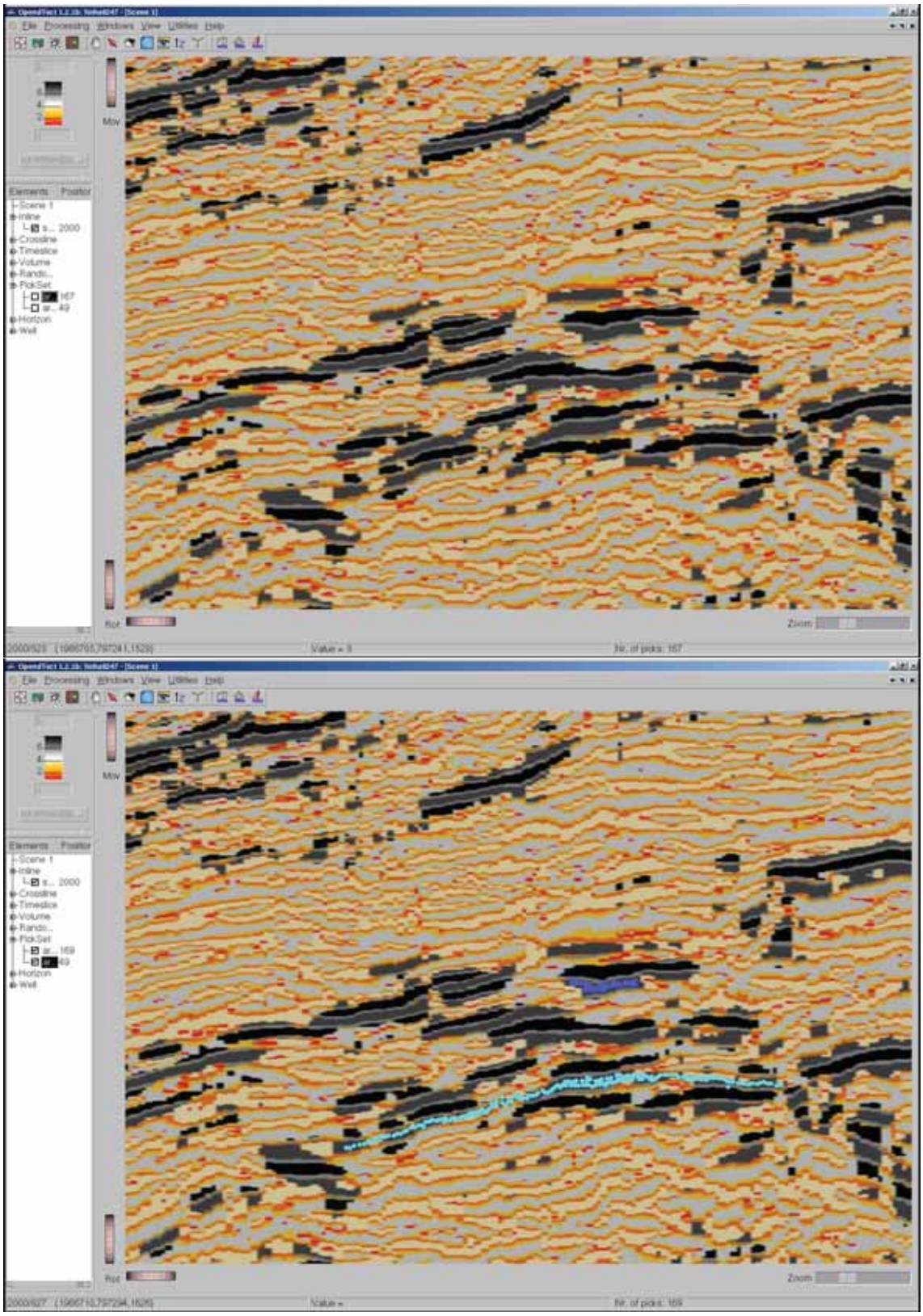


FIGURA 4.5: Janela Sísmica - Melhor Resultado - Rede UVQ

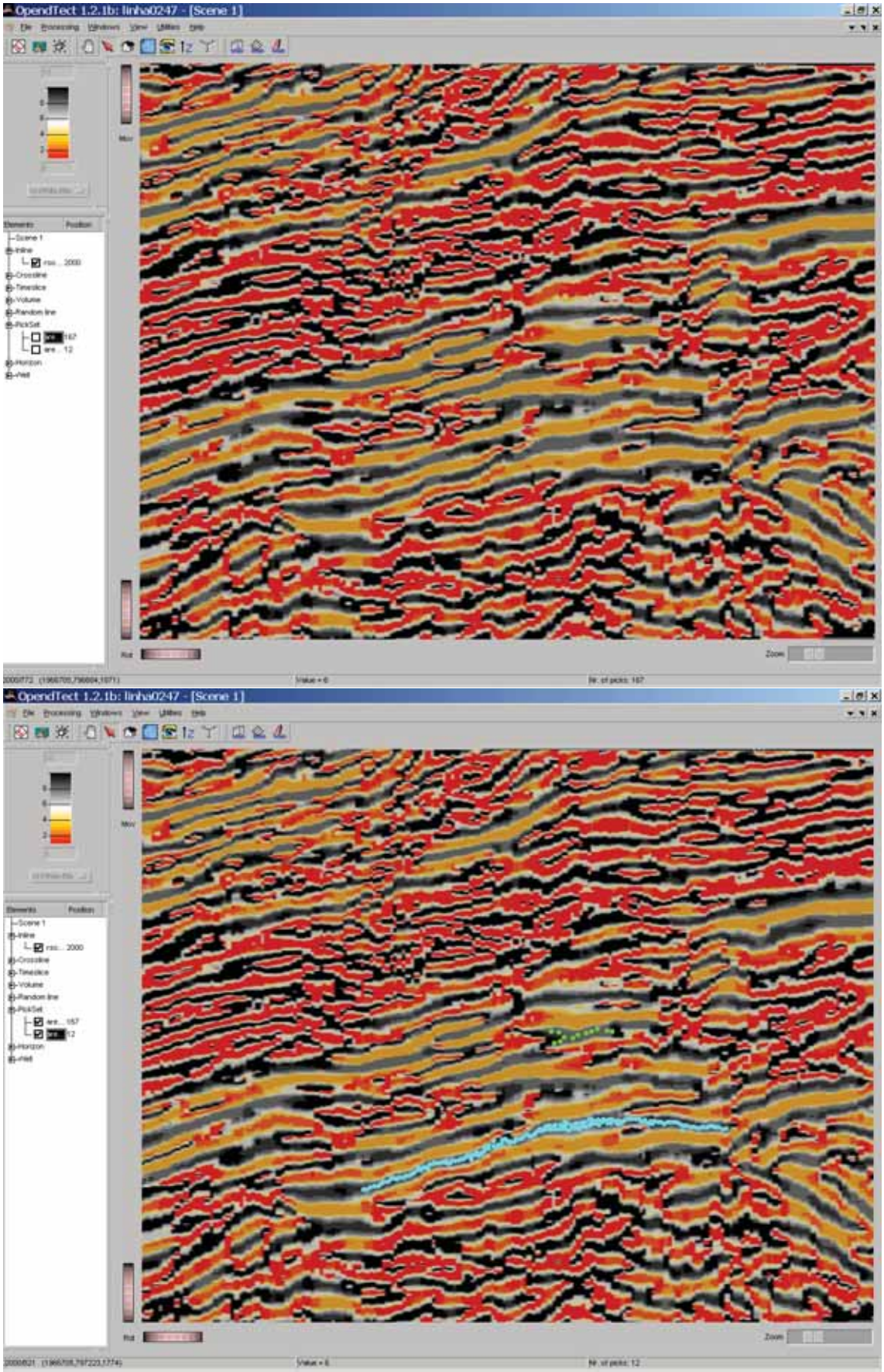


FIGURA 4.6: Janela Sísmica - Melhor Resultado - Rede SOM

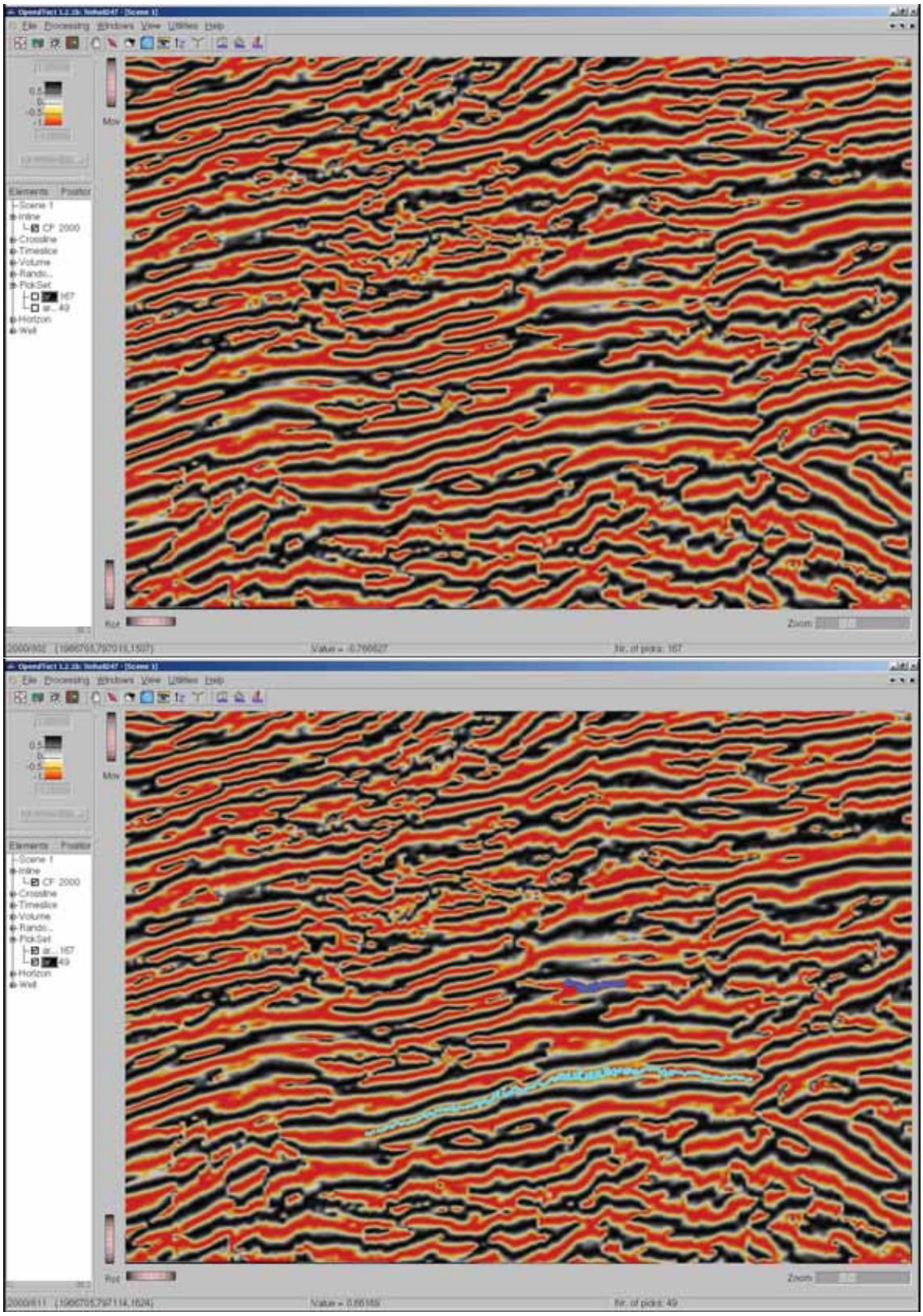


FIGURA 4.7: Janela Sísmica - Atributo Sísmico Cosseno da Fase

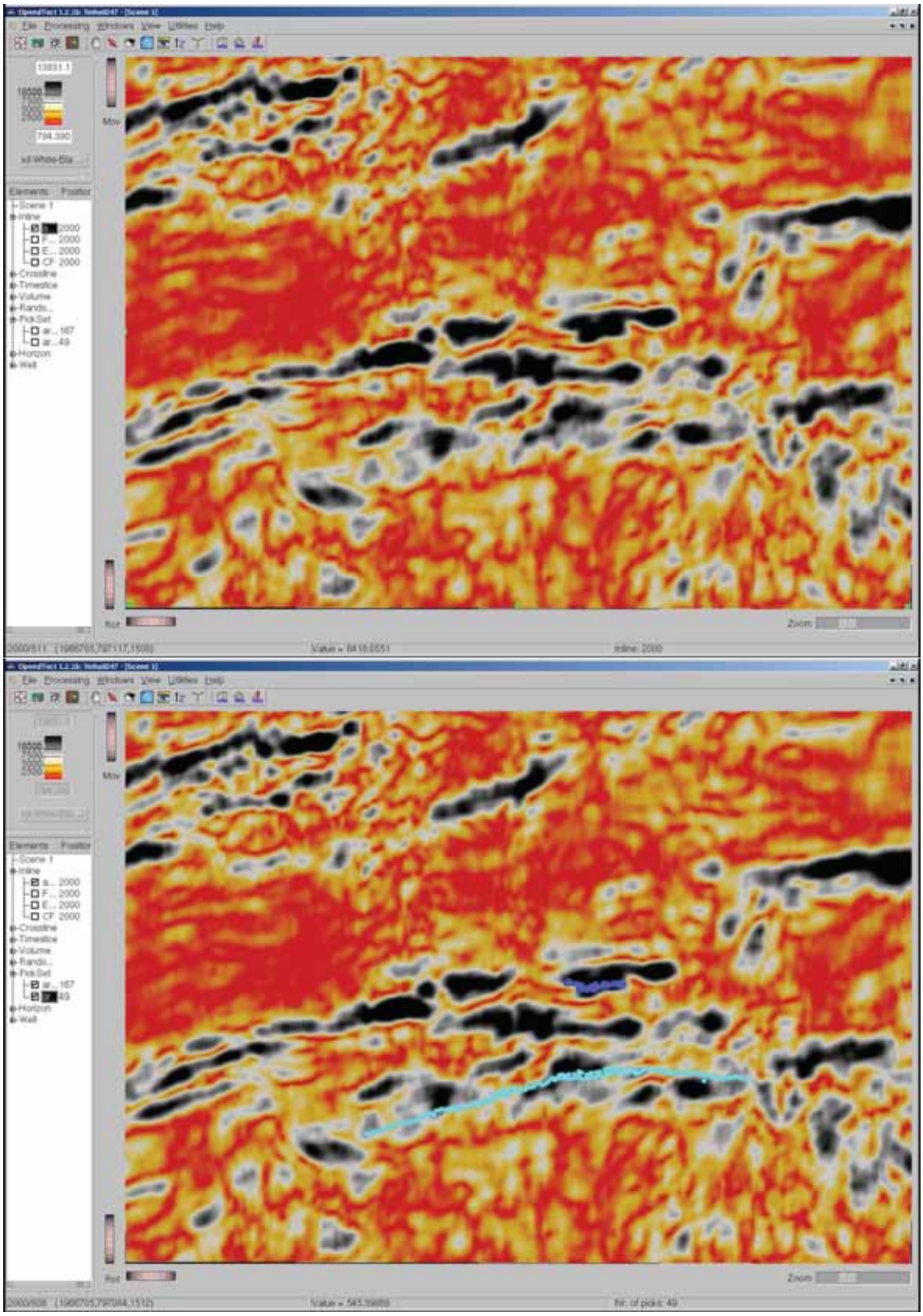


FIGURA 4.8: Janela Sísmica - Atributo Sísmico Amplitude

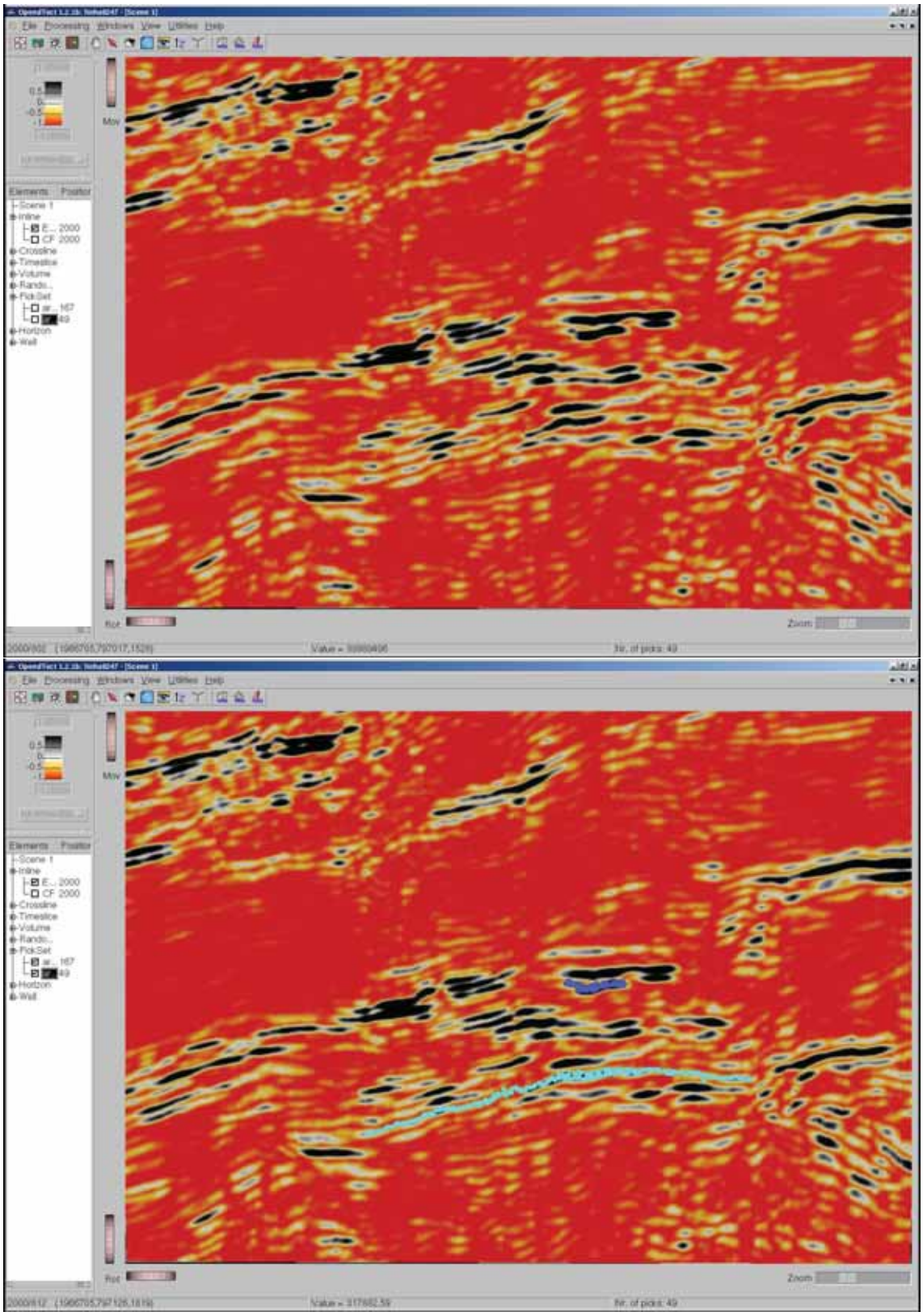


FIGURA 4.9: Janela Sísmica - Atributo Sísmico Energia

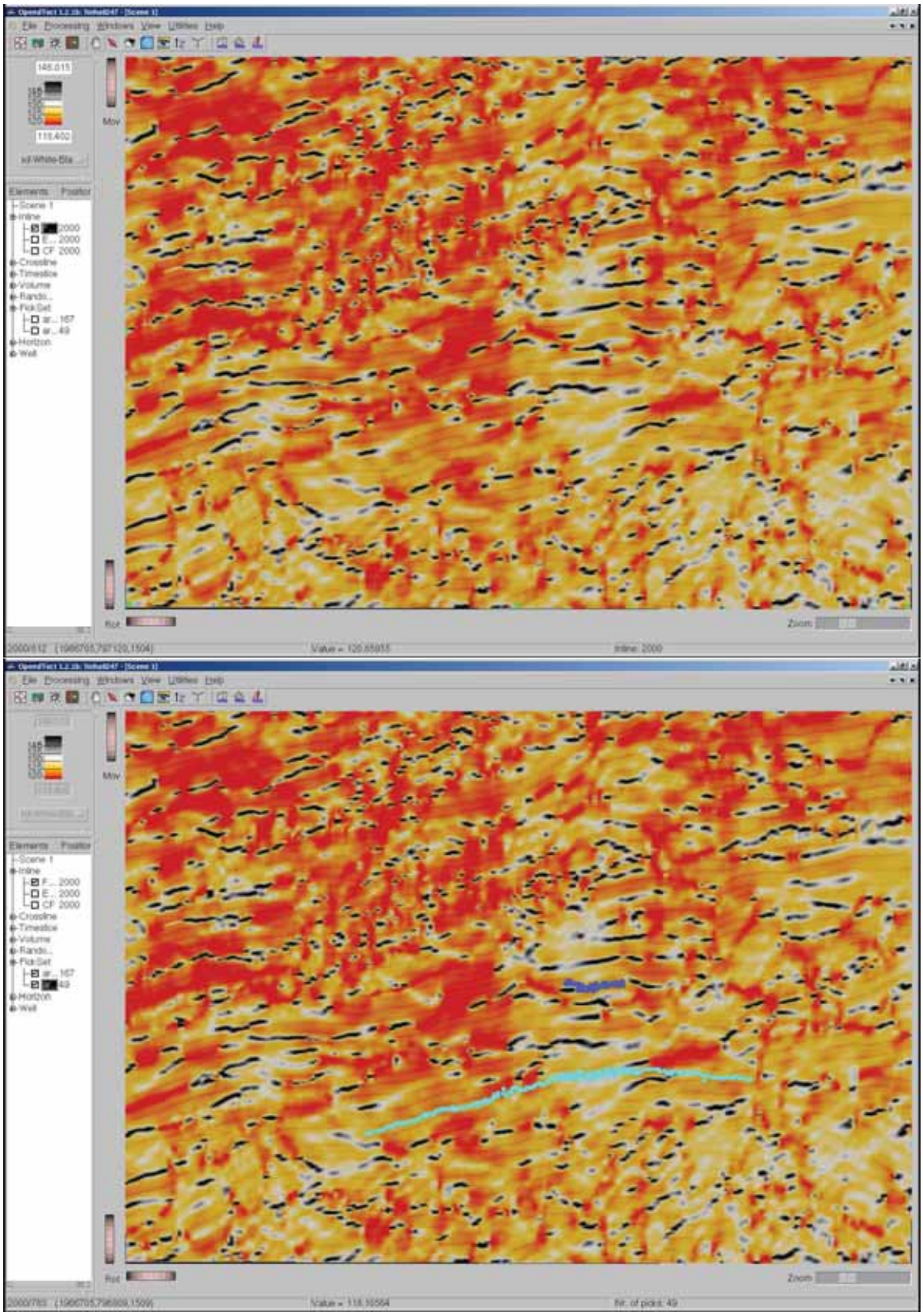


FIGURA 4.10: Janela Sísmica - Atributo Sísmico Freqüência Média

Capítulo 5

Conclusões e perspectiva de trabalhos futuros

A maioria das aplicações voltadas para pesquisas de hidrocarbonetos utiliza base de dados mais completas comparadas à base de dados utilizada nesta tese (exemplo: [5, 14]).

Em caracterização de reservatórios, são utilizados volumes sísmicos (sísmica 3D) ou estes unidos a registros de poços como bases de dados utilizadas em pesquisas mais recentes. Em modelagem de bacias são utilizadas linhas 2D ou volumes 3D. A ANP cedeu um conjunto de dados contendo volumes sísmicos e registros de poços referentes ao Campo Escola de Namorado, mas estes dados eram insuficientes para que experiências como a apresentada neste trabalho pudessem ser realizadas. Segundo o conhecimento de especialista, não havia como "amarrar" um poço a uma linha sísmica e, conseqüentemente, utilizar perfis elétricos como o de raios gama no auxílio da identificação litológica nestes conjuntos de dados.

Foi utilizado um conjunto de dados cedidos à Universidade Federal do Rio de Janeiro pela ANP, contendo cinco linhas sísmicas e três perfis de poços, o que tornou a utilização de perfis e linhas sísmicas inviável, devido à pouca quantidade de perfis. Neste conjunto de dados, havia o conhecimento de especialistas sobre a localização de pacotes de areias. As areias apresentam indícios de acumulação de óleo e, por isso, identificar um padrão sísmico que as localizasse ao longo de uma linha sísmica, por exemplo, possivelmente auxiliaria especialistas em uma possível identificação de reservatórios de petróleo.

Este trabalho teve seu enfoque na aplicação e comparação de diferentes ferramentas e métodos de agrupamento de dados. Esta foi realizada através da aplicação do índice de

validação de agrupamentos PBM, que indicou o melhor agrupamento em cada algoritmo.

Algumas rotinas e funções foram implementadas no programa Matlab enquanto outros programas foram também utilizados. As redes neurais utilizadas correspondem a rede competitiva, UVQ ($8, 40E + 14$) e SOM ($5, 29E + 15$). A rede competitiva faz parte da caixa de ferramentas do programa Matlab. A rede UVQ faz parte do programa OpendTect. Este programa também é responsável pela exibição de atributos sísmicos e resultados dos agrupamentos de dados no formato de janela sísmica. A rede SOM faz parte do programa SOM_PAK. O algoritmo FCM foi implementado no programa Matlab, assim como o índice de validação PBM. Rotinas e funções conhecidas por SegyMAT foram utilizadas para importar e exportar dados do OpendTect para o Matlab e vice-versa. Outras funções e rotinas foram criadas para auxiliar o treinamento, aplicação e outras tarefas.

Segundo o índice de validação de agrupamentos PBM, os melhores resultados correspondem a oito agrupamentos na rede UVQ e a dez agrupamentos na rede SOM. O resultado da rede UVQ foi exibido em forma de janela sísmica juntamente com os atributos sísmicos utilizados nos conjuntos de dados. A janela sísmica que corresponde ao melhor resultado exibe detalhes não encontrados nas janelas dos atributos sísmicos em separado.

Não se pode afirmar que agrupamentos representem areias. Entretanto, pode-se afirmar que o resultado acima garante, segundo o índice de validação PBM, uma qualidade de agrupamento. A partir deste momento, um especialista poderia analisar estes dados na certeza de que os agrupamentos não foram realizados sem uma validação.

5.1 Perspectiva de trabalhos futuros

Algumas sugestões são apresentadas para futuros trabalhos:

- Uma sugestão é que outros atributos sísmicos possam ser utilizados produzindo-se novas experiência na tentativa de identificar diferentes litologias.

- Uma última sugestão refere-se ao estudo mais aprofundado dos dados cedidos pela ANP. A figura 5.1 exibe uma linha sísmica ("0096-0230"), paralela à linha "0247-5619". Esta linha cobre uma área próxima a um Canyon e esta é uma informação valiosa na pesquisa de hidrocarbonetos. A existência de um Canyon permite a formação de areias turbidíticas, bastante associadas à formação de reservatórios de petróleo. Seria

interessante tentar identificar um padrão sísmico que permitisse localizar os pacotes de areia na linha "0247-5619" e aplicar métodos de agrupamentos nos dados referentes as duas linhas sísmica. Obviamente cada método deve possuir a mesma configuração na aplicação das duas bases de dados, ou seja, os mesmos atributos sísmicos, na tentativa de se descobrir se esta areia é ou não turbidítica.

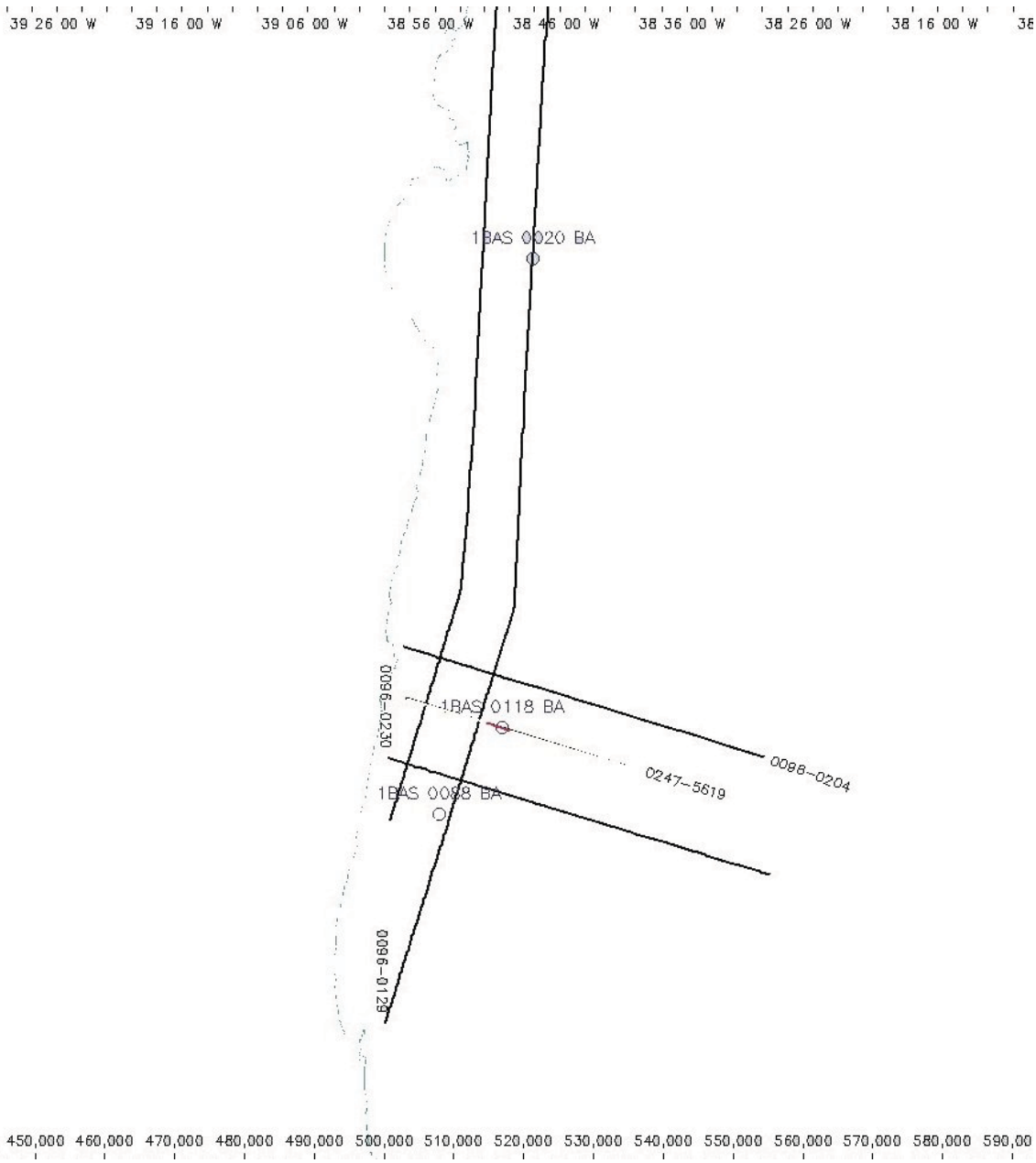


FIGURA 5.1: Localização das linhas sísmicas e perfis de poços

Apêndice A

Rotinas e funções utilizadas na fase de aplicação e validação de agrupamentos

Rotina "Agrup_inicio":

```
%Agrup_inicio
importa_janela;
clear all;
pack;
load entradan;
[o, p] = size(entradan);
if o > p
    entradan = entradan';
end
load treinamenton;
[o, p] = size(treinamenton);
if o > p
    treinamenton = treinamenton';
    [o,p] = size(treinamenton);
end

load entrada;
[o, p] = size(entrada);
```

```
if o > p
    entrada = entrada';
end
load treinamento;
[o, p] = size(treinamento);
if o > p
    treinamento = treinamento';
    [o,p] = size(treinamento);
end

numepocas = 10000;
pi =5;
pf = 15;
for i = pi: pf,

% ---- Rede Competitiva - Treinamento e Simulacao ---
    fprintf(1,'Rede Competitiva : %d Clusters \n',i);
    [u1,net] = agrup_comp(treinamento,entrada,i,numepocas,o);
    save (['u1_comp',num2str(i)],'u1');
    save (['rcomp',num2str(i)],'net');
    clear u1;
% ---- Fuzzy C-Means - Treinamento e Simulacao ---
    fprintf(1,'Fuzzy C-Means : %d Clusters \n',i);
     m = 1.1;
     aleat = 0;
     [u1,w1] = agrup_fcm(treinamenton,entradan,i,m, aleat);
    save (['u1_fcm',num2str(i)],'u1','w1');
    clear u1;
end

aplica_validacao;
```

Rotina "Importa_janela":

```
%Importa Janela Sismica e Prepara Dados para Processamento
[amp, B, C, D] = ReadSegy('amp_sub.segy');
cf = ReadSegyFast('cf_sub.segy');
fmedia = ReadSegyFast('fmedia_sub.segy');
energia = ReadSegyFast('energia_sub.segy');
fprintf('Inicializacao\n');
tic
numAtrib = 4; %Numero de Atributos
[o, p] = size(amp);
%o = numero de linhas e p, de colunas da Matriz Data
entrada = zeros(numAtrib,o*p);
k = 1;
toc
fprintf('Criando Matriz de Entrada de Dados\n');
for i = 1:o,
    for j = 1:p,
        entrada(1,k) = cf(i,j);
        entrada(2,k) = energia(i,j);
        entrada(3,k) = fmedia(i,j);
        entrada(4,k) = amp(i,j);
        k = k+1;
    end
 end
pack;

fprintf('Normalizacao dos dados da Matriz de Entrada\n');
load entrada;
[o,p] = size(entrada);
if o < p
    entrada = entrada';
```

```
    [o,p] = size(entrada);
end

Xmin = zeros(1,p);
Xmax = zeros(1,p);
for i = 1 : p,
    Xmin(1,i) = min(entrada(:,i));
    Xmax(1,i) = max(entrada(:,i));
end

entradan = zeros(o,p); for i = 1:o
    for j = 1:p,
        entradan(i,j) =
        (entrada(i,j) - Xmin(1,j))/ (Xmax(1,j)-Xmin(1,j));
    end
end

save entrada entrada
save entradan entradan

fprintf('Normalizacao dos Dados de Treinamento\n');

load treinamento;
[o,p] = size(treinamento);
if o < p
    treinamento = treinamento';
    [o,p] = size(treinamento);
end

treinamenton = zeros(o,p); for i = 1:o
    for j = 1:p,
        treinamenton(i,j) =
```

```
        (treinamento(i,j) - Xmin(1,j))/ (Xmax(1,j)-Xmin(1,j));
    end
end

save treinamenton treinamenton;
save Xmin Xmin;
save Xmax Xmax;
```

Função "Agrup_comp"

```
%function [u1] = agrup_som(trein,dados,nc,numepocas,numAtributos)
% trein = dados de treinamento
%dados = data set
%nc = numero de clusters
%numepocas =
%numero de epocas utilizadas no treinamento de redes neurais
%numAtributos = numero de Atributos presentes no vetor de entrada

function [u1, rcomp] =
agrup_comp(trein,dados,nc,numepocas,numAtributos) nentrada = [];

for i = 1 : numAtributos,
    nentrada = [nentrada; min(trein(i,:)) max(trein(i,:))];
end
tic
rcomp = newc(nentrada,nc);
rcomp.trainParam.epochs = numepocas;
rcomp = train(rcomp,trein);
s = sim(rcomp,dados);
u1 = vec2ind(s);
toc
return;
```

Função "Agrup_fcm"

```
%function [u1] = agrup_fcm(x,y,nc,m)
% x = dados de treinamento
%y = data set
%nc = numero de clusters
%m = parametro de forma da funcao de perticencia

function [u1,w1] = agrup_fcm(x,y,nc,m,gna)

[ndat,nx] = size(x);
if ndat < nx
    x = x';
    [ndat,nx] = size(x);
end

if gna == 0,
    % inicializacao da matriz de pertinencia
    u1 = rand(ndat,nc);
    u1 = u1./(sum(u1')'*ones(1,nc));
    % inicializacao dos centros de cluster
    su1 = sum(u1 .^ m);
    w1 = (x'*u1.^m)./(ones(nx,1)*su1);
    save (['u1w1_fcm' num2str(nc)], 'u1', 'w1');
else
    load (['u1w1_fcm' num2str(nc)]);
end

%w1 = ones(nx,nc)/2;

% inicio do processo iterativo
fprintf('Treinamento\n');
```

```
tol = 0.0001;
eps1 = inf;
eps2 = inf;

iter = 0;
fprintf('Iteracoes : ');
while eps1 > tol | eps2 > tol
    err = 0;

   w0 = w1;
   u0 = u1;

   su0 = sum(u0.^m);

   w1 = (x'*u0.^m)./(ones(nx,1)*su0);

   for t = 1:ndat
      for j = 1:nc
         d0(j) = 1/distancia(x(t,:)',w1(:,j))^(1/(m -1));
      end

      sd0 = sum(d0);

      for j = 1:nc
         u1(t,j) = d0(j)/sd0;
      end

      err = err + sd0/ndat;
   end

   eps1 = norm(w1 - w0);
   eps2 = norm(u1 - u0);
```

```
   iter = iter + 1;
   if mod(iter,50) == 0,
       fprintf('%d - ',iter);
   end
end
fprintf('Aplicacao\n');
[ndat,natrib] = size(y);
if ndat <
natrib
    y = y';
    [ndat,natrib] = size(y);
end

u1 = zeros(ndat,nc);
for t = 1:ndat
      for j = 1:nc
         d0(j) = 1/distancia(y(t,:)',w1(:,j))^(1/(m -1));
      end
      sd0 = sum(d0);
      for j = 1:nc
         u1(t,j) = d0(j)/sd0;
      end
end

return;
```

Função "distancia"

```
function d0 = distancia(x0,c0);

% distancia euclideana
```

```
d0 = sum((x0 - c0).^2);
```

Rotina "Aplica_Validacao"

```
%aplica_validacao
clear all;
close all;
pi = 5; %numero de particoes inicial
pf = 15; % numero de particoes final
load entrada;
load xmin;
load xmax;

PBM = [];
for j = pi:pf
        load (['rcomp' num2str(j)]);
        load (['u1_comp' num2str(j)]);
        fprintf('Clusters %d - Algoritmo Rede Competitiva \n',j);
        w1n = net.IW{1,1};
        [o,p] = size(w1n);
        w1 = zeros(o,p);
        for k = 1:o
           for m = 1:p,
               w1(k,m) =
               w1n(k,m) * (Xmax(1,m) - Xmin(1,m)) + Xmin(1,m);
           end
        end

        pbm = validacao(entrada,u1,w1,j);
        PBM = [PBM; pbm];
end
```

```
save('pbm_comp','PBM');


PBM = []; for j = pi:pf
    load (['u1_fcm' num2str(j) '_2']);
    fprintf('Clusters %d - Algoritmo Fuzzy C-Means\n',j);
    w1n = w1';
      [o,p] = size(w1n);
        w1 = zeros(o,p);
        for k = 1:o
           for m = 1:p,
               w1(k,m) =
               w1n(k,m) * (Xmax(1,m) - Xmin(1,m)) + Xmin(1,m);
           end
        end
    w1 = w1';
    pbm = validacaof(entrada,u1,w1,j);
    PBM = [PBM; pbm];
    save('pbm_u1_fcm', 'PBM');
end


PBM = []; for j = pi:pf
    load (['u1_uvq' num2str(j)]);
    load (['w_uvq' num2str(j)]);
    fprintf('Clusters %d - Algoritmo UVQ\n',j);
    w1 = w1';
    pbm = validacao(entrada,u1,w1,j);
    PBM = [PBM; pbm];
    save('pbm_u1_uvq', 'PBM');
end
```

Função "validacao"

```
%validacao
function pbm = validacao(x0,u1,w1,nc);
    fprintf('Inicializacao\n');
    [ndados,nvar] = size(x0);
    if ndados < nvar
        x0 = x0';
        [ndados,nvar] = size(x0);
    end
    [d1,d2] = size(u1);
    if d1 > d2
        u1 = u1';
    end
    fprintf('Calculo de coordenadas de toda a base de dados\n');
    centro1grupo = mean(x0); %Media de coordenadas de toda a base

    e1reg = zeros(ndados,1);

    for i=1:ndados
        e1reg(i) = norm(x0(i,:) - centro1grupo);
    end
    fprintf('Calculo de E1\n');
    E1 = sum(e1reg);
    fprintf('Calculo de EK\n');
    distk = zeros(nc,1);
    fprintf('Registros Processados: ');
    d0 = zeros(nc,1);
    for t = 1:ndados
        if mod(t,1000000) == 0
            fprintf('%d - ',t);
        end
```

```
            distk(u1(t)) = distk(u1(t)) + norm(x0(t,:)-w1(u1(t),1));
        end

    EK = sum(distk)
    dinter = [];

    for i = 1 : nc
        for j = 1:nc
            dinter(i,j) = norm(w1(i,:) - w1(j,:));
        end
    end

    %INDICE PBM
    fprintf('Calculos de DK e PBM\n');
    DK = max(max(dinter))
    pbm = ((1/nc)*(E1/EK)*DK)^2
    return;
```

Função "validacaof"

```
%validacaof
function pbm = validacaof(x0,u1,w1,nc);
    fprintf('Inicializacao\n');
    [ndados,nvar] = size(x0);
    if ndados < nvar
        x0 = x0';
        [ndados,nvar] = size(x0);
    end
    [d1,d2] = size(u1);
    if d1 > d2
        u1 = u1';
    end
```

```
fprintf('Calculo de coordenadas de toda a base de dados\n');
centro1grupo = mean(x0); %Media de coordenadas de toda a base

e1reg = zeros(ndados,1);

for i=1:ndados
    e1reg(i) = norm(x0(i,:) - centro1grupo);
end
fprintf('Calculo de E1\n');
E1 = sum(e1reg);
save E1 E1;
fprintf('Calculo de EK\n');
distk = zeros(nc,1);
fprintf('Registros Processados: ');
d0 = zeros(nc,1);
for t = 1:ndados
    if mod(t,1000000) == 0
        fprintf('%d - ',t);
    end
    for j = 1:nc,
        distk(j) =
        distk(j) + ((norm(x0(t,:)'-w1(:,j))) * u1(j,t));
    end
end

EK = sum(distk)
save EK EK;
dinter = [];

for i = 1 : nc
    for j = 1:nc
        dinter(i,j) = norm(w1(:,i) - w1(:,j));
```

```
        end
    end

    %INDICE PBM
    fprintf('Calculos de DK e PBM\n');
    DK = max(max(dinter))
    save DK DK;
    pbm = ((1/nc)*(E1/EK)*DK)^2
    save pbm pbm;
    return;
```

Função "gera_matpart"

```
%gera_matpart_uvq
clear all; close all;

filename = 'suvq_';
for i = 2 : 15
    fprintf('Importacao -  Clusters %d\n',i);
    x = ReadSegyFast([filename num2str(i) '.segy']);
    [namostras,ntracos] = size(x);

    u1 = zeros(namostras*ntracos,1);
    o = 1;
    for j = 1 : namostras
        for k = 1 : ntracos
            u1(o) = x(j,k);
            o = o + 1;
        end
    end
    save (['u1_uvq' num2str(i)],'u1');
end
```

Rotina "matpartsom"

```
%matpartsom
clear all; pack;
pi = 5; pf = 20;
filenames = {'w1h';'w1r'};
filenames2 = {'h';'r'};
[qac m] = size(filenames);
for i = 1 : 2,
    for j = pi : pf
        if i == 1,
            fp = fopen(['w1h' num2str(j) '.out'],'r+')
        else
            fp = fopen(['w1r' num2str(j) '.out'],'r+')
        end
        frewind(fp)
        fprintf(fp,'%c','%');
        st = fclose(fp);
    end
end


load entradan.mat;
[numPontos,numAtrib] = size(entradan);
tic
for i = 1:qac,
    if i == 1,
        fprintf('\nTopologia Hexagonal\n');
    else
        fprintf('Topologia Retangular\n');
    end
```

```
    fprintf('Agrupamentos: ');
    for j = pi:pf,
        fprintf('%d\t',j);
        u1 = zeros(numPontos,1);
        w1 = load([filenames{i} num2str(j) '.out']);
        [K, numAtrib] = size(w1);
        qc = zeros(K,1);
        for r = 1 : numPontos,
            dist = zeros(K,1);
            for k = 1 : K
                dist(k) = distancia(entradan(r,:),w1(k,:));
            end
            u1(r) = find(dist == min(dist));
            qc(u1(r)) = qc(u1(r)) + 1;
            end
        save(['rsom' filenames2{i} num2str(j)],'u1','w1','qc');
        end
end
toc
```

Rotina "aplica_val_som"

```
%aplica_val_som;
clear all; close all;
load entrada;
load xmin;
load xmax;

pi = 5; pf = 20;
PBM = zeros(pf-pi,1);
j = 1;
for i = pi:pf,
    load (['rsomh' num2str(i)]);
```

```
    w = zeros(size(w1));
    [o,p] = size(w1);
    for k = 1:o
        for m = 1:p,
            w(k,m) = w1(k,m) * (Xmax(1,m) - Xmin(1,m)) + Xmin(1,m);
        end
    end
    PBM(j) = validacao(entrada,u1,w,i);
    j = j + 1;
end

save pbm_rsomh PBM;

PBM = zeros(pf-pi,1);
j = 1;
for i = pi:pf,
    load (['rsomr' num2str(i)]);
    w = zeros(size(w1));
    [o,p] = size(w1);
    for k = 1:o
        for m = 1:p,
            w(k,m) = w1(k,m) * (Xmax(1,m) - Xmin(1,m)) + Xmin(1,m);
        end
    end

    PBM(j) = validacao(entrada,u1,w,i);
    j = j + 1;
end

save pbm_rsomr PBM;
```

Referências Bibliográficas

[1] J. E. Thomas, *Fundamentos de Engenharia de Petróleo*. Interciência, 2001.

[2] O. L. S. Corrêa, *Petróleo. Noções sobre Exploração, Perfuração, Produção e Microbiologia*. Interciência, 2003.

[3] J. R. Canuto, "Página do Instituto de Geociências da Universidade de São Paulo." http://www.igc.usp.br/geologia/petroleo.php.

[4] A. B. Strapasson, A. S. Neto, A. T. Soletto, and C. R. F. Brighenti, "Página sobre Geologia do Petróleo." http://geocities.yahoo.com.br/geologiadopetroleo/.

[5] E. M. Fleck, C. E. Pedreira, and R. Santos, "Agrupamentos de dados sísmicos através do algoritmo de kohonen," *VI Congresso Brasileiro de Redes Neurais*, 2003.

[6] T. Taner, "Attributes Revisited." http://www.rocksolidimages.com/pdf/attrib_revisited.htm, Setembro 2000.

[7] M. Nikravesh and M. Hassibi, "Intelligent Reservoir Characterization," *Journal of Petroleum Science and Engineering*, vol. 29, 2001.

[8] G. Bacoccoli and I. C. Toffoli, "Petrobrás and brazil´s offshore exploration: 20 years - a review," *20th Annual Offshore Technology Conference*.

[9] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik, "Validity index for crisp and fuzzy clusters," *Pattern Recognition*, June 2004.

[10] A. de Pádua Braga, T. B. Ludemir, and A. C. P. de Leon Ferreira Carvalho, *Redes Neurais Artificiais, Teoria e Aplicações*. LTC, 2000.

[11] S. Haykin, *Redes Neurais, Princípios e Prática*. Bookman, 2ª ed., 2001.

[12] F. M. de Azevedo, L. M. Brasil, and R. C. L. de Oliveira, *Redes Neurais com aplicações em Controle e em Sistemas Especialistas*. Visual Book, 2000.

[13] L. Fausett, *Fundamentals of Neural Networks. Architectures, Algorithms and Applications*. Prentice Hall, 1994.

[14] M. Nikravesh, F. Amizadeh, and L. Zadeh, *Soft Computing and Intelligent data analysis in oil exploration*. Elsevier, 2003.

[15] P. M. Wong, F. X. Jian, and I. J. Taggart, "A critical coparison of neural networks and discriminant analysis in lithofacies, porosity and permeability predictions," *Journal of Petroleum Geology*, vol. 18, no. 2, pp. 191–206, 1995.

[16] T. Kohonen, *Self-Organizing Maps*. Springer, 2001.

[17] P. F. M. Groot, "Artificial Neural Networks." http://www.dgb-group.com/upload/files/publications/articles/neural/NN_intro.pdf.

[18] P. F. M. Groot, *Seismic reservoir characterisation employing factual and simulated wells*. PhD thesis, Delft Uviversity of Technology.

[19] J. Leski, "Towards a robust fuzzy clustering," *Fuzzy Sets Syst.*, vol. 137, no. 2, pp. 215–233, 2003.

[20] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algoritms*. Plenum Press, 1981.

[21] Élia Yathie Matsumoto, *Matlab6, Fundamentos de Programação*. Érica, 2001.

[22] D. Hanselman and B. Littlefield, *Matlab6, Curso Completo*. Prentice Hall, 2004.