UNIVERSIDADE FEDERAL DA BAHIA
INSTITUTO DE GEOCIÊNCIAS
CURSO DE GRADUAÇÃO EM GEOFÍSICA

GEO213 – TRABALHO DE GRADUAÇÃO

PROPOSIÇÃO DE METODOLOGIA DE INVERSÃO SÍSMICA USANDO REDES NEURAIS

NAYGUEL DE CASTRO COSTA

SALVADOR – BAHIA

JULHO - 2010

Proposição de Metodologia de Inversão Sísmica usando Redes Neurais

por

Nayguel de Castro Costa

GEO213 – TRABALHO DE GRADUAÇÃO

Departamento de Geologia e Geofísica Aplicada

do

Instituto de Geociências

da

Universidade Federal da Bahia

Comissão Examinadora

_____ Dr. Wilson M. Figueiró - Orientador

_____ Dra. Jacira C. B. de Freitas

_____ Dr. Amin Bassrei

Data da aprovação: 16/07/2010

Dedico este trabalho às três
mulheres de minha vida:
Minha mãe, Jô, por sua fé em mim,
minha esposa, Lenísia, por seu amor
verdadeiro,
e minha filha Alice, por fazer minha
vida mais feliz.

RESUMO

Uma rede neural artificial é um processador de informações inspirado no cérebro humano. O elemento chave neste tipo de processador é sua arquitetura, que é composta de um grande número de unidades de processamento (neurônios) fortemente conectadas e trabalhando em união para resolver um problema. Para alcançar os resultados desejados, a rede deve passar por uma etapa de aprendizagem, que consiste na mudança iterativa de parâmetros internos, após a apresentação de vários padrões de entrada, e suas saídas correspondentes.

Neste trabalho, propomos uma metodologia para a solução de um problema de inversão de velocidades sísmicas, utilizando uma rede neural treinada através do algoritmo de retro-propagação do erro. Nós geramos sismogramas sintéticos CSG (*Common-Shot Gathers*) no programa TESSERAL, a partir de modelos unidimensionais de velocidade. Os sismogramas foram parametrizados por série trigonométrica, e os coeficientes dessa parametrização constituíram a entrada da rede neural. Foram apresentados 15 pares (modelo-sismograma) na etapa de treinamento da rede, e reservados outros 5 para testar a capacidade de generalização da mesma.

Após o treinamento a rede não foi capaz de recuperar com exatidão nenhum dos modelos de velocidade, no entanto, se aproximou consideravelmente em alguns dos casos, e em outros, a despeito da inexatidão, conseguiu recuperar informações importantes, tais como: tendência da variação das velocidades com a profundidade e número de camadas. Os piores resultados ocorreram entre os exemplos com características pouco presentes nas amostras de treinamento.

A parametrização de sismogramas demonstrou ser capaz de fornecer as informações necessárias para a obtenção de bons resultados pela rede, embora não tenha sido capaz de recuperar a imagem original do sismograma.

ABSTRACT

An artificial neural network is an information processor inspired by the human brain. The key element in this kind of processor is its architecture, which consists of a large number of processing units (neurons) strongly connected and working together to solve a problem. In order to achieve the desired results, the network must pass through a stage of learning, that is the iterative change of internal parameters, after the presentation of various input patterns, and their corresponding outputs.

In this paper, we propose a methodology for solving a problem of inversion of seismic velocity using a neural network trained by error backpropagation algorithm. From one-dimensional models of velocity, we generate synthetic seismograms CSG (*Common-Shot Gathers*) in the program TESSERAL . Seismograms have been parameterized by trigonometric series, and the coefficients formed the input of the neural network. 15 pairs were presented (model-seismogram) during the network training phase, and booked another 5 to test the ability to generalize from it.

After training, the network was not able to accurately retrieve any of the velocity models, however it was considerably closer in some cases and in others, despite the inaccuracy, it was able to retrieve important information such as trend of the variation in velocity with depth, and number of layers. The worst results occurred among the examples with rare features present in training samples.

The parameterization of seismograms has shown to be able to provide the information necessary to obtain good results over the network, although it has not been able to recover the original image of the seismogram.

ÍNDICE

ÍNDICE DE FIGURAS

INTRODUÇÃO

Recuperar o modelo de velocidades sísmicas a partir de um sismograma registrado consiste em resolver um problema inverso, e é uma tarefa de difícil resolução. Uma das razões que torna complexo o processo de inversão sísmica está na dificuldade de obtenção de modelos matemáticos realísticos que representem com grau máximo de aproximação a distribuição de propriedades sísmicas de subsuperfície. Grande parte dos métodos tradicionais de inversão sísmica são de natureza estatística e dependem de vários passos de tentativa e erro, bem como da intervenção humana a cada passo.

Novos métodos têm sido utilizados para otimizar problemas que envolvem grandes espaços de busca. A Inteligência Computacional (IC) busca, através de técnicas inspiradas na natureza, desenvolver sistemas inteligentes que imitem características humanas, tais como: aprendizado, raciocínio, e adaptação. Dentre os algoritmos mais utilizados na IC, encontram-se as Redes Neurais Artificiais.

Redes neurais são sistemas adaptativos que relacionam um espaço de vetores de entrada com um espaço correspondente de vetores de saída. Visando realizar o relacionamento desejado, a rede passa por um processo de treinamento que consiste no uso de um conjunto de exemplos para, de modo iterativo, produzir mudanças em valores internos de ponderação (pesos), até que se chegue a uma configuração de pesos numéricos que minimize a diferença entre o padrão de saída computado e o desejado, para cada padrão de entrada. Desse modo, a rede torna-se uma aproximação da função que relaciona dados de entrada com aqueles de saída no domínio restrito definido pelos exemplos de treinamento. O fato de uma rede neural ser capaz de recuperar a relação funcional entre um espaço de entrada e um de saída, através do reconhecimento e aproveitamento de padrões, sugere sua aplicação para os casos em que não existe um claro entendimento da física que relaciona um espaço ao outro, ou por ser esta de grande complexidade.

Röth e Tarantola (1991) utilizaram a técnica de redes neurais para obter a velocidade RMS (*Root Mean Square*), tendo sucesso com a rede treinada para modelos de subsuperfície com uma única camada. Röth e Tarantola (1994) aperfeiçoaram o trabalho anterior em uma abordagem em que a rede neural recebe um sismograma CSG (*Common-Shot Gathers*) como entrada e fornece como saída um modelo com 8 camadas de velocidade constante mais o semi-espaço abaixo destas.

Enquanto o problema inverso apresenta dificuldades, o problema direto de gerar sismogramas sintéticos a partir de modelos sísmicos de velocidades em subsuperfície é, em termos

numéricos, relativamente simples, sendo possível gerar um grande número de sismogramas a partir dos modelos propostos sísmicos de subsuperfície. Esses pares (modelo-sismograma) podem constituir uma prática base de dados de treinamento para uma rede neural.

CAPÍTULO 1

Fundamentos Teóricos

Os aspectos teóricos do presente trabalho concentram-se nas especificidades técnicas das chamadas redes neurais, pois estas, em si mesmas, desconsideram aspectos físicos e geológicos das situações estudadas e se restringem apenas àqueles de caráter numérico. Portanto, a questão da parametrização de modelos e dados adquire uma especial importância, pois esta permite que eles sejam tratados como vetores ou matrizes de valores numéricos. Isto é de grande conveniência para a rede neural, pois objetos de grande apelo visual (modelos geológicos e sismogramas), podem ser numericamente tratados.

1.1 Séries Ortogonais com Duas Variáveis e o Polinômio Trigonométrico

Em Kreider et al. (1972) podemos encontrar a apresentação e a demonstração da seguinte proposição:

Teorema 1 *Sejam $\{f_i(x)\}$ e $\{g_j(y)\}$ bases ortogonais dos espaços euclidianos CP[a,b] e CP[c,d], respectivamente. Então, o conjunto de todos os produtos $\{f_i(x).g_j(y)$, i e $j \in \mathbb{N}\}$ é uma base de CP(R), onde R é o retângulo [a,b] $\times$ [c,d] e $CP(R)$ é o conjunto das funções contínuas por partes definidas em R.*

A partir desse teorema é possível encontrar os coeficientes da série:

$$\sum_{i,j=1}^{\infty} \alpha_{ij} f_i(x) g_j(y), \tag{1.1}$$

de qualquer função F de $CP(R)$ através da seguinte expressão:

$$\alpha_{ij} = \frac{\iint_R F(x,y) f_i(x) g_j(y) dR}{\iint_R f_i(x)^2 g_j(y)^2 dR}. \tag{1.2}$$

Santos e Figueiró (2006) particularizaram o teorema citado, a fim de representar modelos de velocidades sísmicas através de uma única função polinomial trigonométrica bidimensonal.

Desse modo, o campo de velocidades sísmicas $V(x,z)$ é representado por:

$$V(x,z) = \sum_{i+j=0}^{N} \alpha_{i,j} f_i(x) g_j(z), \tag{1.3}$$

onde

$$f_i(x) = \frac{1}{2}\left\{[(-1)^i + 1]\cos\left[\frac{iX}{2}\right] + [(-1)^{i+1} + 1]\,\mathrm{sen}\left[\frac{(i+1)X}{2}\right]\right\}, \tag{1.4}$$

$$g_j(z) = \frac{1}{2}\left\{[(-1)^j + 1]\cos\left[\frac{jZ}{2}\right] + [(-1)^{j+1} + 1]\,\mathrm{sen}\left[\frac{(j+1)Z}{2}\right]\right\}, \tag{1.5}$$

$$X = \frac{\pi(2x - l)}{l}, \tag{1.6}$$

$$Z = \frac{\pi(2z - d)}{d}, \tag{1.7}$$

$$\alpha_{i,j} = \frac{\iint_R V(x,z) f_i(x) g_j(z) dR}{\iint_R f_i(x)^2 g_j(z)^2 dR}, \tag{1.8}$$

e R é a região espacial dos pontos $(x,z) \in [0,l] \times [0,d]$, onde o modelo é considerado, e l e d representam, respectivamente, o comprimento e a profundidade máxima do modelo.

Neste trabalho estas expressões serão utilizadas para parametrizar sismogramas, isto é, ao invés de modelos sísmicos de velocidade, isto é, substitui-se $V(x,z)$ por $A(x,t)$. Onde A representa as amplitudes registradas nos receptores na posição x e no tempo t. Por conveniência chamaremos $\mathcal{T}$ de tempo máximo.

1.2 Perceptrons de Múltiplas Camadas e Aprendizagem por Retropropagação

O propósito dessa seção é fornecer ao leitor o vocabulário básico usado para descrever redes neurais de múltiplas camadas que usam o algoritmo de retropropagação de erro em sua fase de treinamento.

1.2.1 Algumas Considerações Preliminares

Redes neurais são sistemas dinâmicos com um grande número de unidades de processamento (UPs) conectadas, chamadas de neurônios. O perceptron é o tipo de estrutura de dados

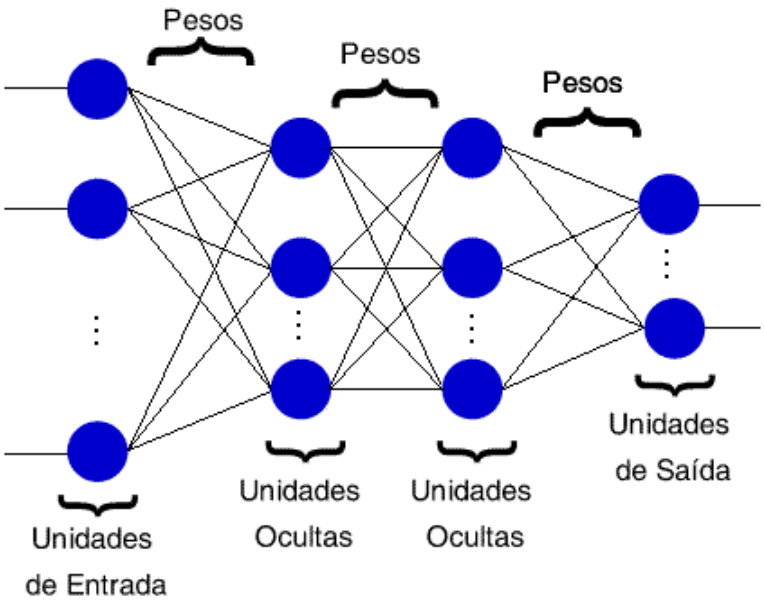


Figura 1.1: Grafo arquitetural de um perceptron de múltiplas camadas com duas camadas ocultas. Fonte: www.lncc.br/ labinfo

mais utilizado para estudar as redes neurais. A Figura 1.1 mostra o grafo arquitetural de um perceptron de múltiplas camadas com duas camadas ocultas e uma camada de saída. Neste trabalho iremos considerar unicamente o caso da rede totalmente conectada. Isto significa que um neurônio em qualquer camada da rede está conectado a todos os neurônios da camada anterior (Haykin, 2001).

Cada neurônio é uma UP, recebendo números reais como entrada e os transformando em um valor de saída. A saída é transmitida por meio de conexões, que possuem um peso definido. Antes de um valor ser transmitido, ele é multiplicado pelo peso correspondente. Modificar os valores dos pesos, por sucessivas aplicações de uma regra de aprendizagem, permite que a rede se aproxime de uma função que relaciona os padrões de entrada às saídas desejadas.

1.2.2 Algoritmo de Retropropagação

O sinal de erro do neurônio j, na iteração n, é definido por

$$e_j(n) = d_j(n) - y_j(n), \tag{1.9}$$

sendo d_j a saída desejada, e y_j o sinal funcional processado no neurônio j.

O valor instantâneo $\mathcal{E}$(n) da energia total do erro é expresso da seguinte forma:

$$\mathcal{E}(n) = \frac{1}{2}\sum_{j=1}^{J} {e^2}_j(n), \tag{1.10}$$

onde J é o número de unidades de saída.

Se N representa o número total de exemplos apresentados durante o treinamento, a energia média quadrática do erro é obtida somando-se os $\mathcal{E}(n)$ para todos os n, e então os normalizando em relação ao tamanho N do conjunto, tal como expresso em:

$$\mathcal{E}_{med} = \frac{1}{N}\sum_{n=1}^{N}\mathcal{E}(n), \tag{1.11}$$

onde $\mathcal{E}(n)$ e $\mathcal{E}_{med}$ são funções de todos os parâmetros livres da rede. O objetivo do processo de treinamento da rede é ajustar os parâmetros livres de modo a minimizar $\mathcal{E}_{med}$. Tal minimização pode ser feita utilizando-se o algoritmo do mínimo quadrado médio (MQM).

A Figura 1.2 representa o neurônio j sendo alimentado por um conjunto de sinais funcionais produzidos pelos neurônios à sua esquerda. Definimos o campo local induzido $v_j(n)$ como:

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n)y_i(n), \tag{1.12}$$

onde m é o número total de entradas aplicadas ao neurônio j. A unidade de bias é adicionada, como sendo simplesmente o peso w_{j0}, já que o bias tem um valor constante de 1.0. O papel do bias será descrito na seção 1.2.3. Assim, o sinal funcional $y_j(n)$ que aparece na saída do neurônio j na iteração n é:

$$y_j(n) = \varphi(v_j(n)) \tag{1.13}$$

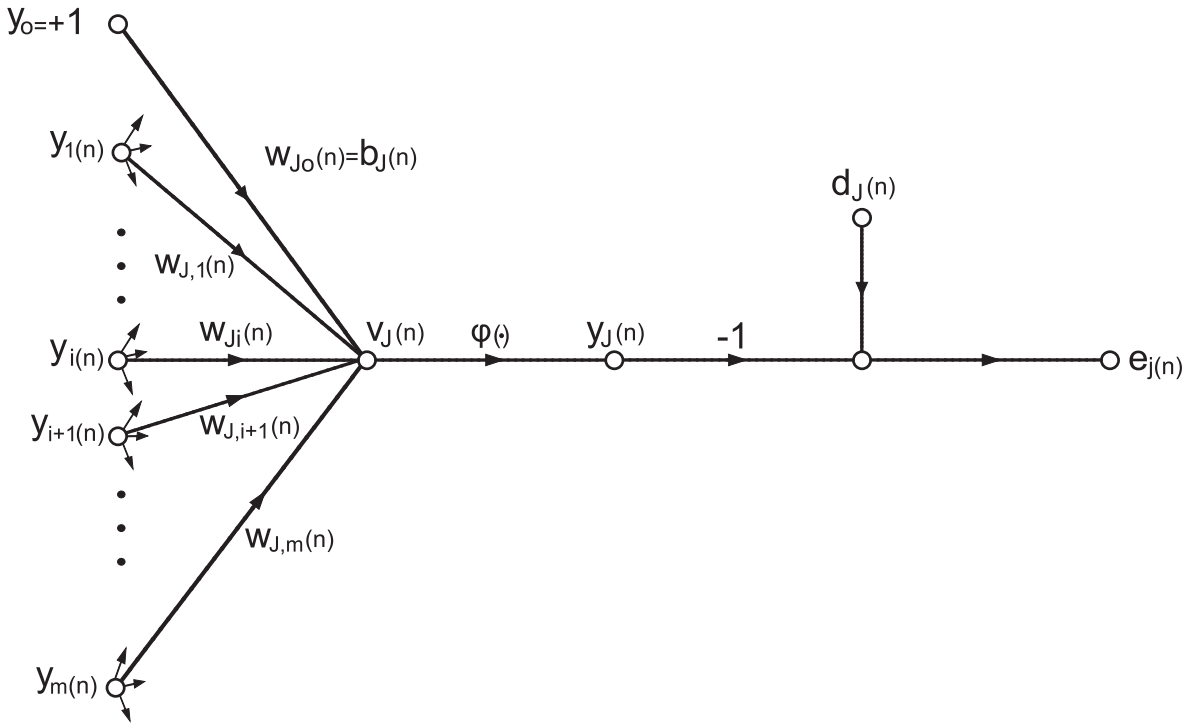


Figura 1.2: Grafo de fluxo de sinal ressaltando os detalhes do neurônio de saída j. Adaptado de HAYKIN, 2001

A correção Δw_{ji}, aplicada ao peso sináptico w_{ji}, deve ser proporcional a derivada parcial $\partial\mathcal{E}(n)/\partial w_{ji}(n)$. De acordo com a regra da cadeia, podemos escrever essa derivada parcial como:

$$\frac{\partial\mathcal{E}(n)}{\partial w_{ji}(n)} = \frac{\partial\mathcal{E}(n)}{\partial e_j(n)}.\frac{\partial e_j(n)}{\partial y_j(n)}.\frac{\partial y_j(n)}{\partial v_j(n)}.\frac{\partial v_j(n)}{\partial w_{ji}(n)} \tag{1.14}$$

Diferenciando-se ambos os lados das Equações (1.10), (1.9), (1.13), e (1.12); obtemos:

$$\frac{\partial\mathcal{E}(n)}{\partial e_j(n)} = e_j(n), \tag{1.15}$$

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1, \tag{1.16}$$

$$\frac{\partial y_j(n)}{\partial v_j(n)} = \frac{d\varphi(v_j(n))}{dv_j(n)}, \tag{1.17}$$

e

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = y_i(n). \tag{1.18}$$

Retornando a Eq. (1.14) chegamos a:

$$\frac{\partial\mathcal{E}(n)}{\partial w_{ji}(n)} = -e_j(n).\frac{d\varphi_j(v_j(n))}{dv_j(n)}.y_i(n). \tag{1.19}$$

Obtemos, então, a assim chamada regra delta:

$$\Delta w_{ji}(n) = -\eta\frac{\partial\mathcal{E}(n)}{\partial w_{ji}(n)} = \eta\delta_j(n)y_i(n), \tag{1.20}$$

onde η é o parâmetro taxa de aprendizagem e δ_j(n) é o gradiente local, que é definido da seguinte forma:

$$\delta_j(n) = -\frac{\partial\mathcal{E}(n)}{\partial v_j(n)} = e_j(n)\frac{d\varphi_j(v_j(n))}{dv_j(n)}. \tag{1.21}$$

O gradiente local aponta para a direção que melhor modifica os pesos sinápticos.

Nota-se que para efetuarmos as correções nos pesos, precisamos conhecer o sinal de erro $e_j(n)$ na saída do neurônio j. Se o neurônio j pertence a camada de saída, é trivial calcular o sinal de erro, pois a saída desejada é conhecida para cada par de treinamento. No caso do

neurônio j pertencer a uma camada oculta, é preciso saber como penalizar ou recompensar o neurônio oculto pela sua parcela de responsabilidade no erro obtido (Haykin, 2001). O sinal de erro deve ser determinado recursivamente, em termos dos sinais de erro de todos os neurônios aos quais o neurônio oculto está diretamente ligado.

Utilizando o índice k para os neurônios de saída e o índice j para o neurônio oculto, aplica-se uma abordagem semelhante à anterior para que se chegue a expressão do gradiente local para um nó oculto:

$$\delta_j(n) = \frac{d\varphi_j(v_j(n))}{dv_j(n)} \sum_k \delta_k(n) w_{kj}(n). \tag{1.22}$$

Conhecendo $\delta_j(n)$ tanto para os nós de saída quanto para os ocultos, podemos efetuar a correção de todos os pesos sinápticos da rede. Após a mudança, nós reaplicamos as entradas e repetimos o processo. O ciclo de aplicar uma entrada, calcular uma saída, computar um erro e modificar os pesos constitui uma iteração da rede. A iteração de todos os elementos da amostra de treinamento constitui uma época.

A aprendizagem termina quando o erro atinge um valor menor que uma tolerância especificada ou quando a rede alcança um número de iterações determinada pelo usuário. Após essa parada, não são mais feitas alterações nos pesos, e a rede passa pela fase de validação, quando são apresentados os novos modelos à rede observando-se a reposta calculada pela mesma. Desse modo é possível avaliar se a rede pode resolver o problema que motivou seu treinamento.

As equações utilizadas para a retropropagação representam uma técnica de gradiente, e a rede neural é suscetível de encontrar os mesmos problemas enfrentados por qualquer algoritmo que utiliza esta técnica. A convergência para um mínimo global não é garantida, o aprendizado pode ser muito lento, e as conexões podem alcançar valores extremos que paralizam o processo de aprendizagem.

1.2.3 Parâmetros da Rede

O usuário tem o controle sobre os seguintes parâmetros em uma rede neural com retropropagação de erro:

1. Número de camadas escondidas
2. Números de UP's escondidas
3. Função de ativação
4. Inicialização dos pesos
5. Taxa de aprendizagem
6. Bias.

Número de Camadas Escondidas

Muitos trabalhos se dedicaram a estabelecer a quantidade ideal de camadas escondidas para que a rede neural resolva o problema proposto, entre eles destacam-se os trabalhos de Bishop (1995), Hecht-Nielsen (1990), Cybenco (1989), e Hornik et al. (1989). A conclusão da maioria dos pesquisadores é que uma camada escondida é suficiente, mas que adicionar uma camada escondida pode, em alguns casos, melhorar a precisão e diminuir o tempo de aprendizagem.

Número de UP's Escondidas

Para Poulton (2003), a representação dos dados de entrada e a estrutura do treinamento são, freqüentemente, muito mais críticos do que o número de UP's escondidas no controle da acurácia dos resultados.

Alguns pesquisadores têm sugerido que o tipo de geometria das entradas e saídas pode ser um balizador do número ideal de UP's escondidas em redes com muito menos nós de saída do que de entrada. Um número pequeno de UP's induz a rede a tomar decisões inadequadas. Um número muito grande de UP's faz com que a rede memorize o grupo de treinamento. Para entender melhor o papel das UP's escondidas, sugerimos a consulta aos trabalhos de German et al. (1992) e Lapedes e Farber (1988).

Função de Ativação

A maioria das implementações dos algoritmos de retropropagação para redes neurais utiliza, ou a função sigmoidal, ou a tangente hiperbólica, para realizar a operação de ativação. Tal função também é denominada por alguns autores (Ludwing Jr. & Montgomery, 2007) de função de transferência, pois ela não permite que os valores dos parâmetros do modelo fiquem fora de seus intervalos de validade.

Neste trabalho utilizamos uma função sigmoidal conhecida como função logística, que na sua forma geral é descrita como:

$$\varphi_j(v_j(n)) = \frac{1}{1 + exp(-av_j(n))}, \tag{1.23}$$

onde $a > 0$ é o parâmetro de inclinação da função sigmóide e $v_j(n) \in (-\infty, +\infty)$. Variando-se o parâmetro a, obtemos funções sigmóides com diferentes inclinações. Entre outras vantagens, essa função é facilmente diferenciável, o que é importante na implementação do algoritmo de retropropagação. Tal derivada é dada por:

$$\frac{d\varphi_j(v_j(n))}{dv_j(n)} = \varphi'_j(v_j(n)) = \frac{aexp(-av_j(n))}{[1+exp(-av_j(n))]^2}, \tag{1.24}$$

ou, simplesmente:

$$\varphi'_j(v_j(n)) = ay_j(n)[1-y_j(n)]. \tag{1.25}$$

Inicialização dos Pesos

Inicializar de forma aleatória os pesos sinápticos é uma boa prática. No entanto, é preciso ter cuidado com o tamanho dos valores atribuídos a eles. Valores altos podem saturar a função logística no início do treinamento e prejudicar os passos seguintes. Uma estratégia interessante, é escolher pesos aleatórios tais que, a magnitude da entrada típica na j-ésima UP seja menor, mas não muito menor, do que a unidade. Fazemos isso atribuindo aos pesos w_{ji} a ordem de $1/k_j$, onde k_j é o número de entradas que alimentam a UP. Utilizamos essa estratégia na concepção de nossa rede.

Taxa de Aprendizagem

O algoritmo de retropropagação fornece uma "aproximação" para a trajetória no espaço de pesos calculada pelo método da descida mais íngreme. Quanto menor for o parâmetro da taxa de aprendizagem, η, menor serão as variações dos pesos sinápticos da rede, de uma iteração para a outra, e mais suave será a trajetória no espaço de pesos (Haykin, 2001). Por outro lado, se fizermos η muito grande, na tentativa de acelerar a aprendizagem, a rede irá se tornar instável. Na prática, a maioria dos pesquisadores escolhe a taxa de aprendizagem pelo método da tentativa e erro.

Bias

Uma unidade de bias é uma UP com um valor constante de saída igual a 1.0, conectada a cada UP em uma camada oculta ou de saída. A essas conexões (bias-UP) é atribuído um peso sináptico passível de correções na etapa de treinamento, como em todas as outras conexões da rede. O bias foi introduzido pela primeira vez por Widrow e Hoff (1960), trabalho no qual eles demonstraram que a adição do bias acelera significativamente a convergência da rede.

CAPÍTULO 2

Metodologia

De certo modo, parte deste trabalho consiste em propor uma metodologia não usual para a resolução do problema de inversão de dados sísmicos. Neste capítulo apresentaremos os passos realizados na elaboração do trabalho, desde a criação dos modelos até a construção da rede neural artificial propriamente dita.

2.1 Definição dos Modelos Geofísicos

A primeira fase do trabalho consistiu na criação de modelos sísmicos da terra. Foram gerados 20 modelos $(1 - D)$ com 5 camadas, planas horizontais, de 500 m de espessura sobre um semi-espaço, conforme mostra a Tabela 2.1. Cada valor na tabela representa a velocidade compressional na camada, em km/s. Usando o artifício de repetir as velocidades de camadas subsequentes, criamos modelos de 2 a 6 camadas (se considerarmos o semi-espaço como uma camada). A menor velocidade entre os modelos é $1,5$ km/s, valor que pode ser atribuído a propagação de ondas P em areias saturadas em água, e a maior, $5,5$ km/s , que está dentro do intervalo típico de velocidades em granitos do embasamento. Procuramos, dentro das limitações que impomos, representar o maior número possível de situações geológicas, de modo a conferir capacidade de generalização à rede.

Uma vez definidos os modelos com variação de velocidade exclusivamente vertical, foram gerados sismogramas sintéticos, usando os pacotes de modelagem sísmica de uma versão *trial* do programa TESSERAL. Os grupos de traços sísmicos foram registrados a partir de uma família de tiro comum (*Common-Shot Gathers*), num levantamento multicanal de lanço simétrico, com 10 receptores de cada lado da fonte, totalizando 21 canais, já que o TESSERAL considera a fonte como um receptor, gerando também o traço *zero-offset*. O sinal é registrado durante $\mathcal{T} = 2,93$ s, amostrado a cada $0,01$ s. Os receptores estão a uma distância de 450 m dos vizinhos, de modo que os dois mais distantes da fonte estão a 4.500 m dela. O modelo resultante é um retângulo com 9 km de largura e 3 km de profuncidade, como exemplificado na Figura 2.1. Configuramos no programa para desconsiderar efeitos provocados por ondas múltiplas e cisalhantes.

Os sismogramas sintéticos e os modelos correspondentes formam os pares de treinamento da

Profundidades em km / Modelos	0,0-0,5	0,5-1,0	1,0-1,5	1,5-2,0	2,0-2,5	2,5-∞
M_1	1,5	1,5	1,5	2,5	2,5	2,5
M_2	2,0	2,0	2,0	3,5	3,5	3,5
M_3	2,0	2,0	2,0	4,0	4,0	4,0
M_4	1,5	1,5	2,5	2,5	3,5	3,5
M_5	2,0	2,0	2,5	2,5	3,0	3,0
M_6	2,0	2,0	2,5	2,5	5,0	5,0
M_7	1,5	2,0	2,5	3,0	3,5	4,0
M_8	1,5	2,5	3,5	4,0	4,5	5,0
M_9	2,0	2,5	3,0	3,5	4,5	5,5
M_{10}	2,0	2,0	2,0	3,0	3,5	4,0
M_{11}	2,0	2,0	2,5	3,0	4,0	4,0
M_{12}	2,0	2,5	3,0	5,0	5,0	5,0
M_{13}	1,5	2,0	2,5	3,0	3,5	3,5
M_{14}	1,5	2,5	2,5	2,0	3,0	3,0
M_{15}	1,5	3,5	2,0	2,0	3,5	3,5
M_{16}	1,5	3,5	3,5	4,5	4,5	5,5
M_{17}	1,8	1,8	1,8	2,7	2,7	2,7
M_{18}	2,1	2,1	3,2	3,2	4,1	4,1
M_{19}	1,8	2,1	2,4	3,1	3,3	3,7
M_{20}	1,5	3,5	4,5	3,5	4,5	3,5

Tabela 2.1: Modelos 1-D de velocidades compressionais em subsuperfície. As velocidades estão em km/s.

rede. Na figura 2.2 são apresentados alguns destes pares.

Mencionamos anteriormente que é desejável que uma rede neural possua capacidade de generalização, isto é, seja capaz de reconhecer padrões que não foram utilizados durante a fase de treinamento. Na realidade, essa generalização funciona como uma espécie de tarefa de interpolação efetuada pela rede. Os 5 últimos modelos da tabela e seus respectivos sismogramas foram separados para verificar essa tarefa da rede, nunca sendo utilizados no processo de treinamento.

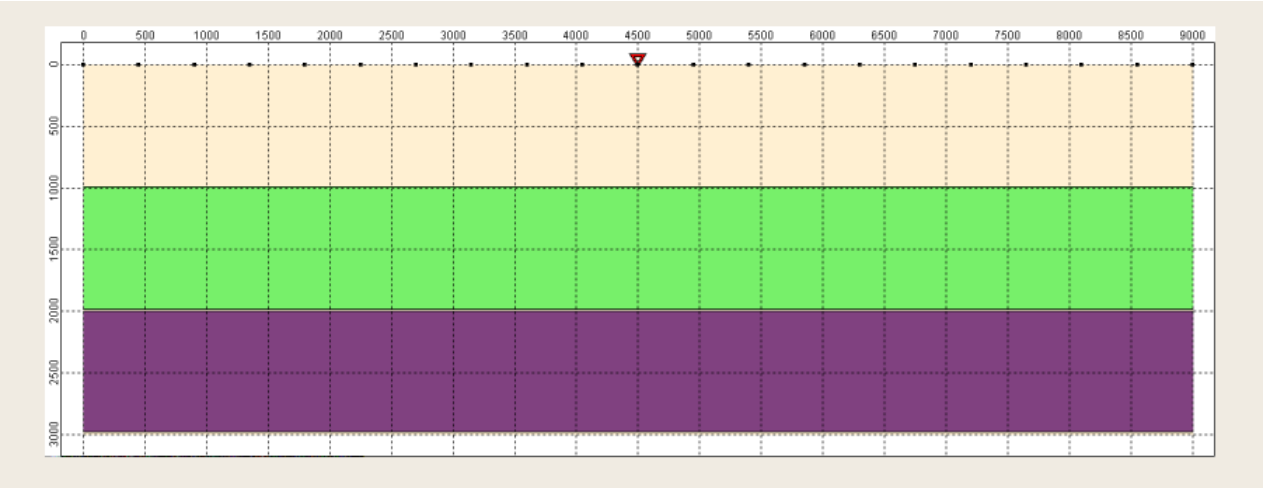


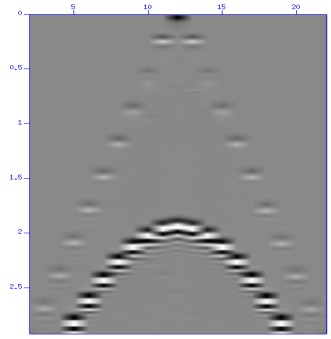
Figura 2.1: Configuração da geometria de aquisição em uma das telas do TESSERAL para o modelo M_5.

2.2 Parametrização Trigonométrica do Sismograma

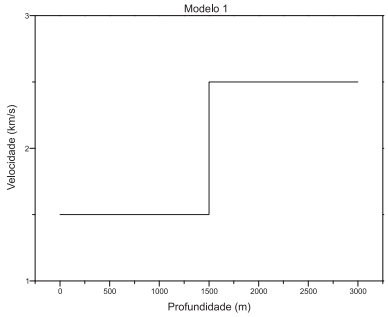
Cada sismograma consiste de 21 traços, amostrados em 293 pontos cada. Desse modo, cada um deles possui $21 \times 293 = 6153$ pontos no tempo. A utilização de uma parametrização possibilita uma drástica redução na quantidade de nós de entrada da rede, simplificando a elaboração e execução do algoritmo de retropropagação do erro, e acelerando o processo de treinamento.

Normalmente as parametrizações são utilizadas para representar modelos geológicos, tanto na resolução do problema direto, como na do problema inverso. A representação de sismogramas por parametrização encontra mais dificuldades do que a de modelos, devido a maior complexidade dos mesmos. Observando as Figuras 2.1 e 2.2(c), podemos prever que a 'imagem' do M_5 é mais fácil de ser recuperada do que a do sismograma correspondente. No entanto, o objetivo de representar um sismograma através de uma parametrização não é o de recuperar as amplitudes originais (embora isso seja desejável e muito interessante), e sim, obter coeficientes que representem de forma única os dados sísmicos originais. Para a rede neural, teoricamente, é indiferente a forma como os dados se apresentam para ela, desde que todos os dados de entrada obedeçam a mesma regra, visando salvaguardar alguma coerência.

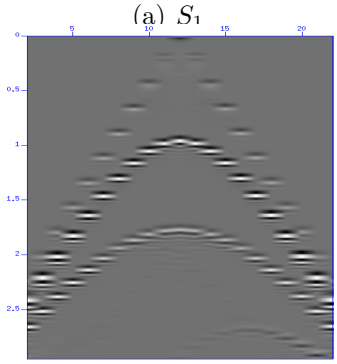
Desse modo, se $M(S)$ é a função hipotética que entrega o modelo quando recebe o sismograma, S, correspondente, e $P(S)$ é outra função hipotética que entrega a parametrização ao receber o sismograma, a rede neural irá aproximar $M(P)$, mas, como P depende de S, ela estará indiretamente aproximando $M(S)$. A princípio, o principal requisito para que isso seja possível é que para qualquer par de sismogramas S_1 e S_2 , $P(S)$ entregue respostas suficientemente diferentes para que a rede reconheça-os como padrões distintos.



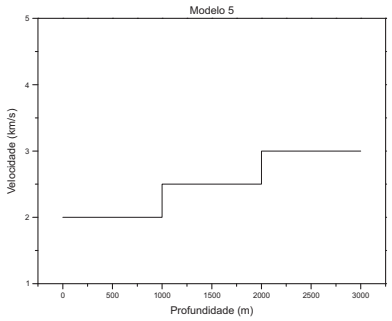
(a) S_1



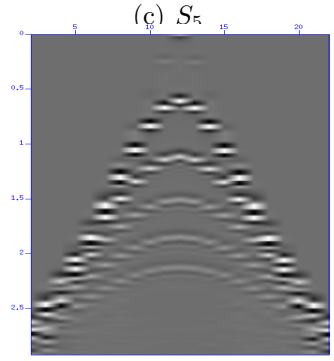
(b) M_1



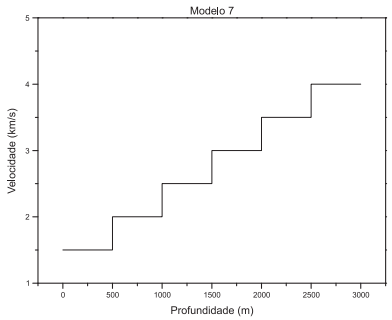
(c) S_5



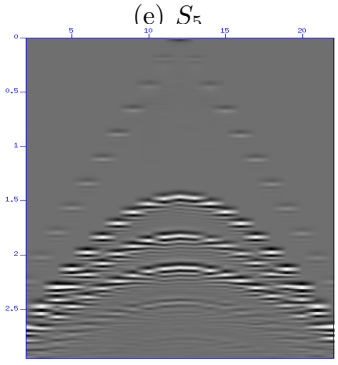
(d) M_5



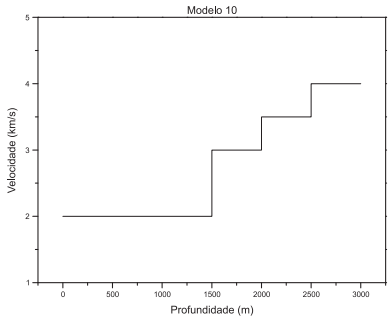
(e) S_5



(f) M_7



(g) S_5



(h) M_{10}

Figura 2.2: Modelos M_1, M_5, M_7 e M_{10} (à direita), e seus respectivos sismogramas S_1, S_5, S_7 e S_{10} (à esquerda) gerados pelo programa de modelagem sísmica TESSERAL

Entre as diversas opções existentes na literatura, escolhemos usar a parametrização por série trigonométrica, discutida na seção 1.1. O que caracteriza a série são seus coeficientes, já que $f_i(x)$ e $g_j(t)$ são as mesmas para todos os sismogramas. No Apêndice A encontram-se os passos para a obtenção da expressão analítica de $\alpha_{i,j}$ através da fórmula:

$$\alpha_{i,j} = \frac{\iint_D A(x,t) f_i(x) g_j(t) dD}{\iint_D f_i(x)^2 g_j(t)^2 dD}, \tag{2.1}$$

onde D é a região dos pontos $(x,t) \in [0;l]\times[0,\tau]$.

Aqui apresentamos a expressão resultante, que foi utilizada na implementação do algoritmo, a saber:

$$\alpha_{i,j} = \frac{4}{\pi^2} \sum_{m=1}^{p} \sum_{n=1}^{q} A_{m,n}. \begin{cases} K_0.M_0 & se \quad i \quad é \quad 0 \quad e \quad j \quad é \quad 0 \\ K_0.M_j & se \quad i \quad é \quad 0 \quad e \quad j \quad é \quad par \\ K_0.N_j & se \quad i \quad é \quad 0 \quad e \quad j \quad é \quad ímpar \\ K_i.M_0 & se \quad i \quad é \quad par \quad e \quad j \quad é \quad 0 \\ L_i.M_0 & se \quad i \quad é \quad ímpar \quad e \quad j \quad é \quad 0 \\ K_i.M_j & se \quad i \quad é \quad par \quad e \quad j \quad é \quad par \\ K_i.N_j & se \quad i \quad é \quad par \quad e \quad j \quad é \quad ímpar \\ L_i.M_j & se \quad i \quad é \quad ímpar \quad e \quad j \quad é \quad par \\ L_i.N_j & se \quad i \quad é \quad ímpar \quad e \quad j \quad é \quad ímpar \end{cases} \tag{2.2}$$

sendo

$$K_0 = \frac{\pi}{p}, \tag{2.3}$$

$$M_0 = \frac{\pi}{q}, \tag{2.4}$$

$$K_i = \frac{1}{i} \left\{ \text{sen} \left[\frac{i\pi}{2p} (2m - p) \right] - \text{sen} \left[\frac{i\pi}{2p} \left(2(m-1) - p \right) \right] \right\}, \tag{2.5}$$

$$L_i = \frac{1}{1+i} \left\{ \cos \left[\frac{(i+1)\pi}{2p} (2(m-1) - p) \right] - \cos \left[\frac{(i+1)\pi}{2p} (2m - p) \right] \right\}, \tag{2.6}$$

$$M_j = \frac{1}{j} \left\{ \text{sen} \left[\frac{j\pi}{2q} (2n - q) \right] - \text{sen} \left[\frac{j\pi}{2q} \left(2(n-1) - q \right) \right] \right\}, \tag{2.7}$$

$$N_j = \frac{1}{1+j} \left\{ \cos \left[\frac{(j+1)\pi}{2q} (2(n-1) - q) \right] - \cos \left[\frac{(j+1)\pi}{2q} (2n - q) \right] \right\}, \tag{2.8}$$

$A_{m,n}$ é a matriz das amplitudes registradas no sismograma, p é o número de traços, e q é o número de amostras temporais por traço.

Utilizamos essas equações para implementar um programa em FORTRAN que calculou os coeficientes de Fourier para cada sismograma sintético gerado anteriormente. Optamos por calcular 45 coeficientes ($i + j$ variando de 0 a 8), assumindo serem eles suficientes para representar os sismogramas.

As Figuras 2.3, 2.4, e 2.5, mostram exemplos de superfícies de amplitudes geradas através da seguinte equação:

$$A(x,t) = \sum_{i+j=0}^{8} \alpha_{i,j}.f_i(x).g_j(t), \tag{2.9}$$

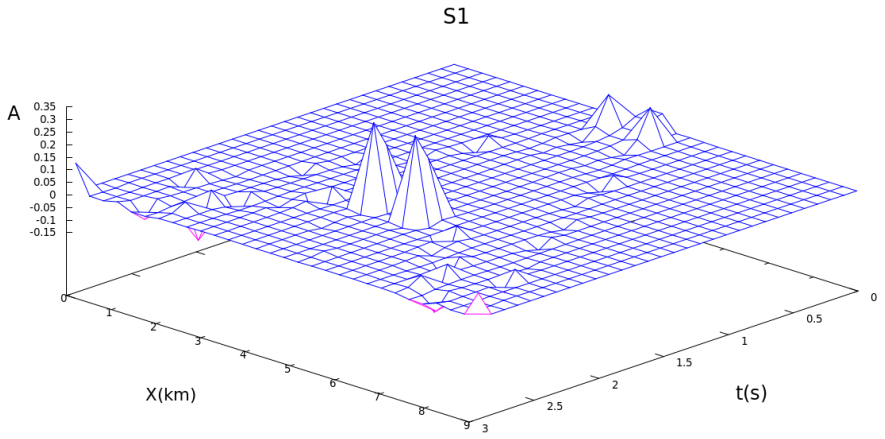
utilizando os 45 $\alpha_{i,j}$'s calculados com a rotina que desenvolvemos, comparadas com as superfícies dos sismogramas originais. Podemos observar que não há uma relação clara de "parentesco" entre os dois tipos de superfícies. Isso era esperado, devido as dificuldades inerentes a complexidade de um sismograma, mencionadas anteriormente no texto. Porém, é importante notar que, assim como percebemos visualmente as diferenças nos padrões das imagens geradas por três sismogramas diferentes (os exemplos apresentados nas figuras são provenientes, propositalmente, de modelos de 2, 3 e 6 camadas, para evidenciar essas diferenças), podemos também observar diferenças entre as superfícies obtidas por parametrização. Admitindo isso, podemos avançar para a próxima etapa do procedimento de inversão.

2.3 Projeto da Rede Neural

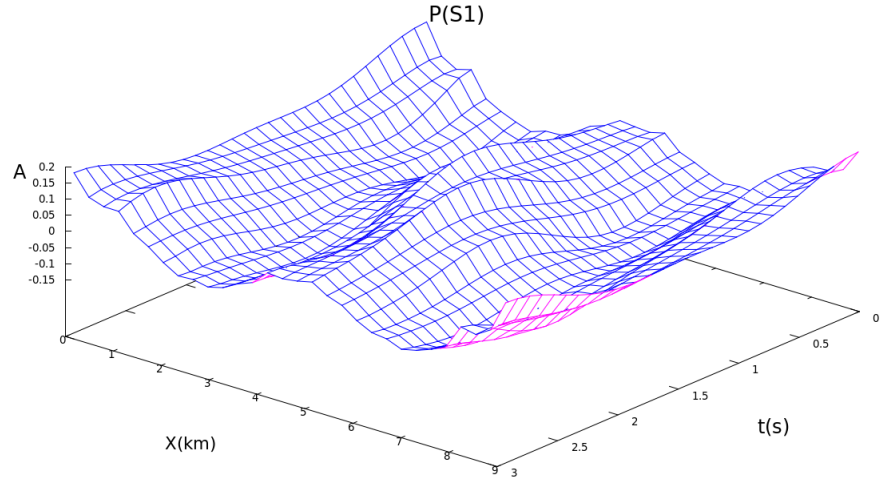
2.3.1 Número de Conexões

Após a parametrização, a rede recebe uma camada de entrada com 45 neurônios. Optamos por construir uma rede com uma única camada escondida com 25 neurônios. A escolha do número de UP's escondidas foi de certa forma arbitrária, respeitando somente o critério de possuir um número inferior a 45 (para que a rede não 'decorasse' os exemplos de treinamento) e grande o suficiente para que a rede possua parâmetros livres suficientes para a aproximação correta da função. Sendo assim, nossa rede tem sua arquitetura representada na Figura 2.6.

Nossa rede neural é completamente conectada, em outras palavras, todos os neurônios de uma camada se comunicam com os da camada seguinte, o que faz com que tenhamos um total de $46 \times 26 \times 6 = 7176$ conexões. O número adicionado aos neurônios de entrada e ocultos é devido ao bias. Os 7176 pesos sinápticos são os parâmetros livres da rede.



(a)



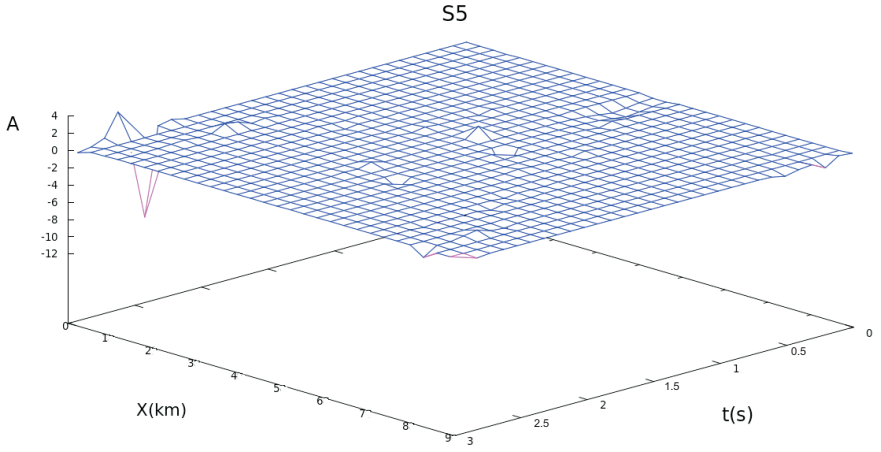
(b)

Figura 2.3: (a) Sismograma original: S_1. (b) Superfície gerada com os coeficientes da parametrização do sismograma S_1.

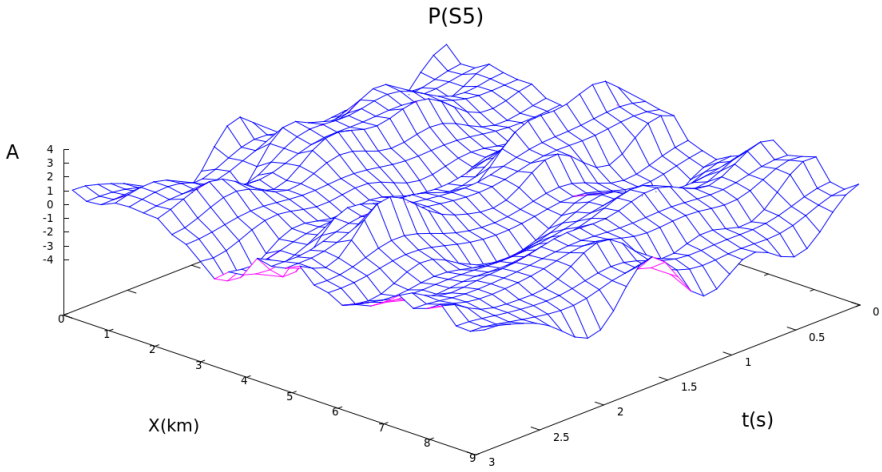
2.3.2 Representação dos Dados de Entrada e Saída

Duas questões fundamentais precisam ser respondidas antes de iniciar a construção da rede: "Como eu irei representar meus dados de entrada e saída e quantos exemplos de treinamento eu irei precisar?" (Poulton, 2003). A segunda questão não pode ser adequadamente respondida até que a primeira o seja.

Muitas técnicas de pré-processamento podem ser aplicadas aos dados de entrada e de saída, porém, é preciso estar atento ao custo computacional. Um dos grandes benefícios da aplicação de redes neurais é prover um meio extremamente rápido para processar dados. No entanto, se o pré-processamento demandar altos custos operacional e computacional, perderemos grande parte do tempo que ganho no processamento propriamente dito, não justificando a utilização desta ferramenta.



(a)



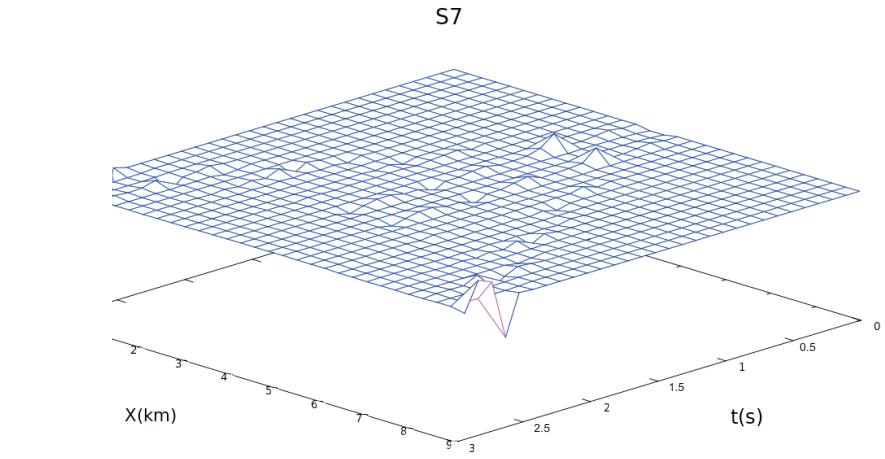
(b)

Figura 2.4: (b) Sismograma original: S_5. (b) Superfície gerada com os coeficientes da parametrização do sismograma S_5.

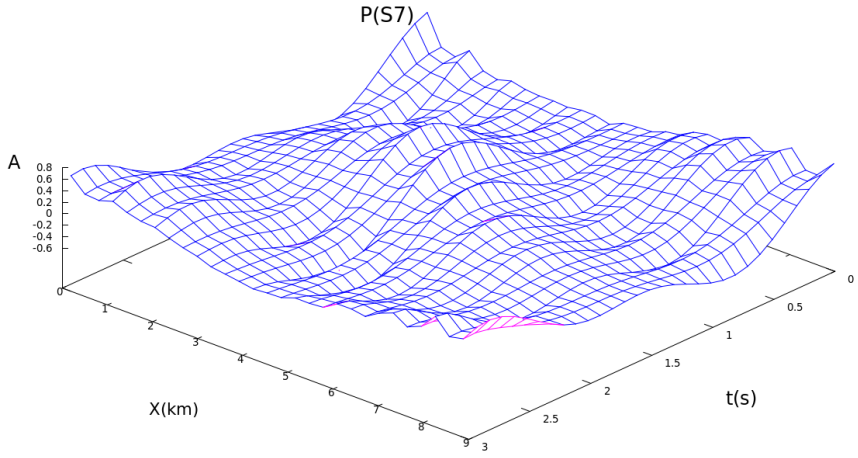
A única técnica de pré-processamento utilizada em nosso projeto foi a mudança de escala, motivada pelo fato dos coeficientes da série trigonométrica possuirem valores baixos, da ordem de 10^{-3}, o que os faz pouco capazes de sensibilizar a rede, se aplicados diretamente. Utilizamos, então, a seguinte mudança de escala linear:

$$x_i^{esc} = \frac{(r_{max} - r_{min})(x_i - d_{min})}{d_{max} - d_{min}} + r_{min}, \tag{2.10}$$

onde as variáveis d_{max} e d_{min} representam a faixa de valores para cada conjunto de treinamento e r_{max} e r_{min} representam a faixa desejada de valores, que depende da função de ativação. Aplicando a Equação (2.12) aos coeficientes da série, iremos garantir que todos eles estarão dentro da faixa de valores desejada, e que, principalmente, todos eles (exceto os nulos) irão contribuir para o ajuste dos pesos da rede. No caso da funções sigmóides a faixa mencionada é [0,1].



(a)



(b)

Figura 2.5: (a) Sismograma original: S_7.(b) Superfície gerada com os coeficientes da parametrização do sismograma S_7.

2.3.3 Estimativa do Tamanho do Grupo de Treinamento

Uma rede neural deve ser treinada usando vários exemplos. A escolha da quantidade de exemplos de nosso grupo de treinamento se deu mais devido às limitações enfrentadas na geração dos sismogramas, do que na observância de algum tipo de critério. O processo de geração de dados sísmicos no TESSERAL é, de certa forma, 'artesanal' e demorado. Além disso, por se tratar de uma versão *trial*, não tivemos tempo de gerar muitos modelos.

Através de uma investigação matemática detalhada, Baum e Haussler (1989), estabeleceram qual deve ser a relação entre o número de exemplos de treinamento versus o tamanho da rede, para se obter uma rede treinada que generaliza corretamente uma certa fração de novos padrões de entrada. De acordo com eles, redes com uma camada escondida devem satisfazer

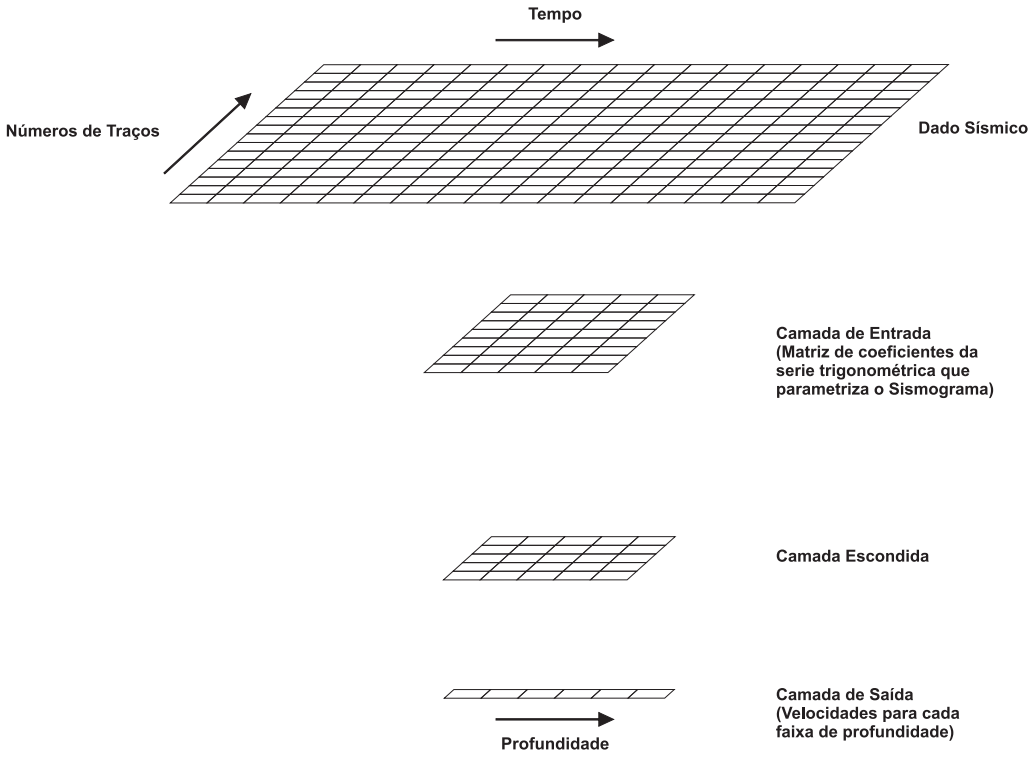


Figura 2.6: A rede neural utilizada em nosso experimento. A camada de entrada tem 45 neurônios, a escondida 25, e a de saída 6.

a relação $\frac{n}{C}$, onde n é o número de pesos e 1-C a fração de generalizações corretas. Aplicando a arquitetura de nossa rede, se quísessemos ter um percentual de acerto de generalizações de 90% , precisaríamos de cerca de 70.000 exemplos de treinamento. Comparar esse número com os nossos 15 exemplos mostra que, mesmo trabalhando em desvantagem, nossos resultados (que serão apresentados na próxima seção) estão longe de serem considerados ruins.

Röth e Tarantola (1994), mostraram experimentalmente, que uma rede que realiza razoavelmente bem a generalização dos exemplos, pode ser obtida com um número significativamente menor de exemplos do que o previsto por Baum e Hassler. Eles conseguiram excelentes resultados com 450 exemplos de treinamento, quando o valor teórico para obter resultados semelhantes seria da ordem de 1.300.000. Baseado nesse resultado, e no fato de nosso trabalho consistir de uma proposta semelhante à deles, podemos ter esperança de conseguir bons resultados com nossos poucos exemplos.

2.3.4 Critério de Parada

Não é possível demonstrar que o algoritmo de retropropagação convergiu, de modo a encerrar as iterações no momento ideal. Ao invés disso, existem alguns critérios razoáveis que podem ser usados para encerrar o ajuste dos pesos. Uma boa maneira de escolha de critério de parada, é utilizar uma das propriedades únicas dos mínimos locais ou globais da superfície

de erro.

Sabemos que se um vetor de peso w^* representa um mínimo, seja ele local ou global, a função de custo $E_{med}(w)$ é estacionária em $w = w^*$. Desse modo, utilizamos o seguinte critério de convergência:

> Considera-se que o algoritmo de retropropagação tenha convergido quando a taxa absoluta de variação do erro médio quadrado por época for suficientemente pequena (Haykin, 2001).

A taxa de variação de 1% do erro médio quadrático foi considerada suficientemente pequena em nosso algoritmo.

2.3.5 Resumo do Algoritmo da Rede Neural

Levando-se em conta as considerações mencionadas nos itens anteriores, implementamos o programa do perceptron de múltiplas camadas para o treinamento de nossa rede em FORTRAN. Resumimos os passos do algoritmo, na forma de fluxograma na Figura 2.7. Um aspecto importante ainda não mencionado é que os exemplos de treinamento devem ser apresentados, preferencialmente, de forma aleatória em cada época, de modo a obter melhores resultados. Respeitamos isso na elaboração do programa.

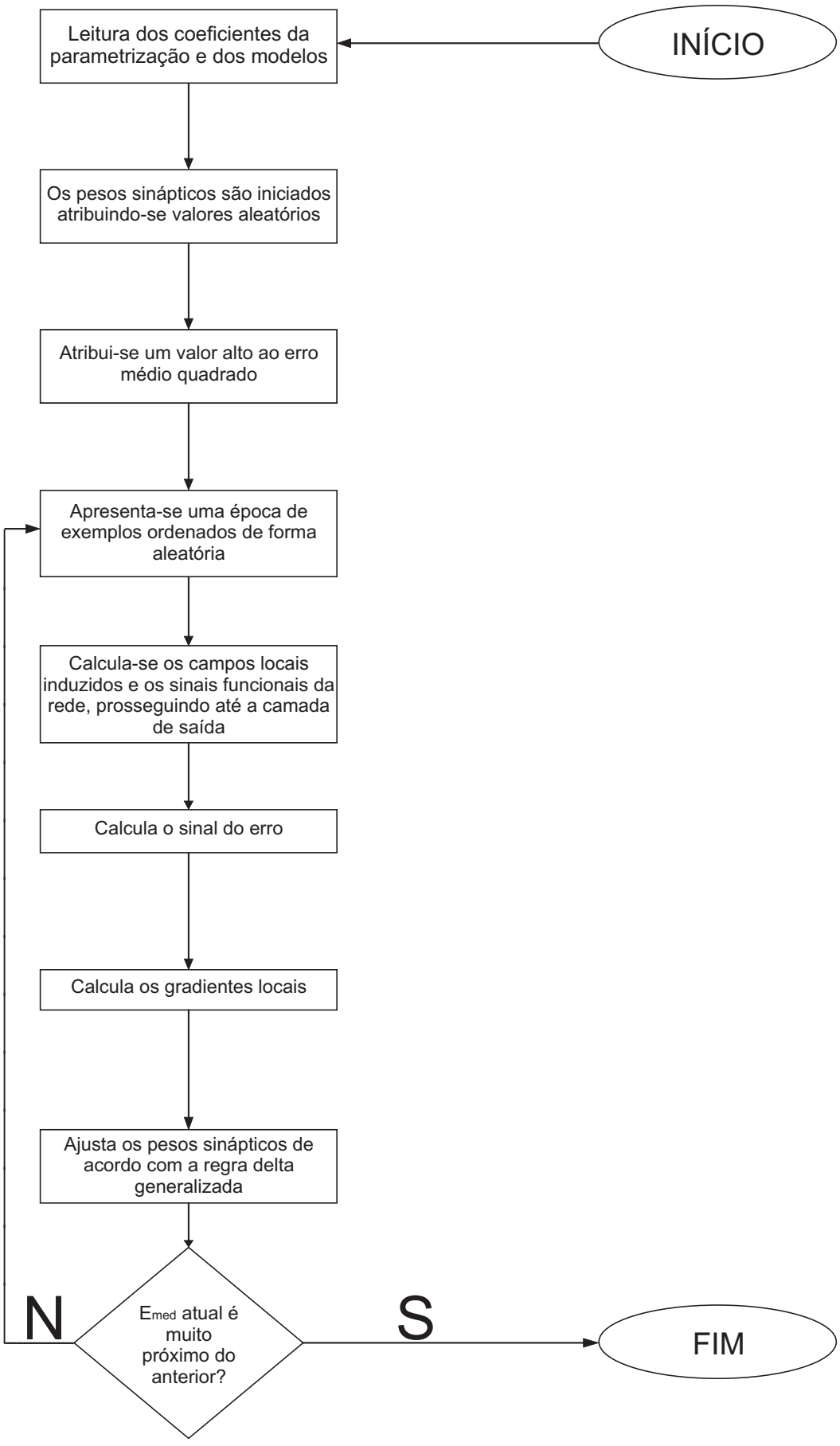


Figura 2.7: Fluxograma que representa o algoritmo de treinamento da rede neural.

CAPÍTULO 3

Resultados Após o Processo de Treinamento

A rede atingiu nosso critério de parada após 223 iterações. Com o término do treinamento, 'congelamos' os pesos sinápticos e aplicamos novamente os 15 modelos do treinamento. Em seguida utilizamos os cinco sismogramas restantes, que não participaram do processo de treinamento para testar a capacidade de generalização da rede.

3.1 Recuperação dos Modelos Sísmicos do Grupo de Treinamento

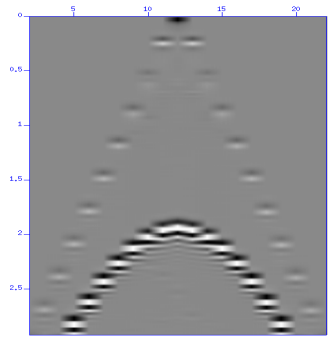
O lado esquerdo da Figura 3.1 mostra 4 sismogramas típicos utilizados na fase de treinamento, e no lado direito da mesma figura vemos, em linhas pretas, o modelo sísmico "real", que chamaremos de agora em diante de "desejado", pois o objetivo da rede é obtê-lo como resposta. Isto é, a figura apresenta os seguintes pares modelo-sismograma usados no treinamento da rede: (M_1,S_1), (M_5,S_5), (M_7,S_7) e (M_{10},S_{10}).

As linhas vermelhas nas representações gráficas dos modelos apresentados na Figura 3.1 correspondem a saída da rede, para estes 4 exemplos mencionados. A taxa de aprendizagem η escolhida foi $0,3$; foram realizados testes com η igual a 0.2; 0.4; e 0.5; mas o valor que trouxe melhores resultados foi o primeiro. O valor do parâmetro de inclinação, a, da função de ativação utilizado foi $1,73$; valor recomendado por Haykin (2001).

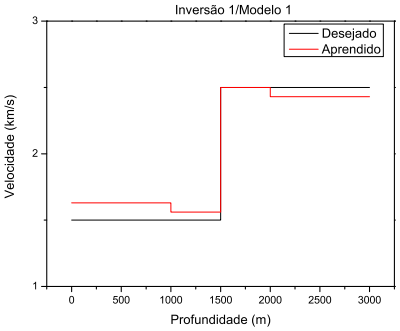
Embora a rede não tenha sido capaz de recuperar o modelo sísmico original de forma exata, ela obteve uma boa acurácia, exceto no caso modelo M_5 e de todos os demais modelos de 3 camadas. Essencialmente, a figura mostra que uma rede neural como a nossa é capaz de inverter muitos dos sismogramas que são apresentados a ela na etapa de treinamento.

A rede demonstrou ter a tendência de entregar na saída respostas iguais para modelos mais parecidos, como pode ser visto na Figura 3.2. Neste caso ela respondeu aos dois sismogramas com as velocidades do modelo M_1.

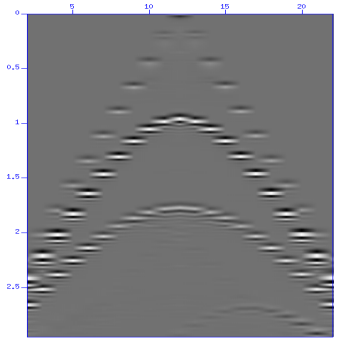
Os modelos de M_{11} a M_{15} são mais complexos que os 10 primeiros, inclusive dois deles apresentam camadas de baixa velocidade. Nesse caso os resultados não são bons como podemos ver nos dois exemplos apresentados na Figura 3.3.



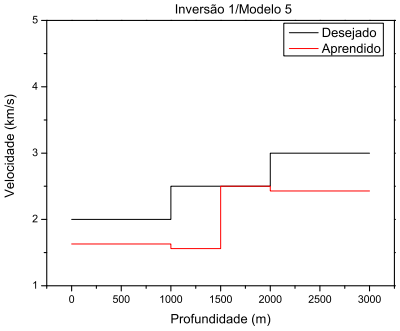
(a) S_1



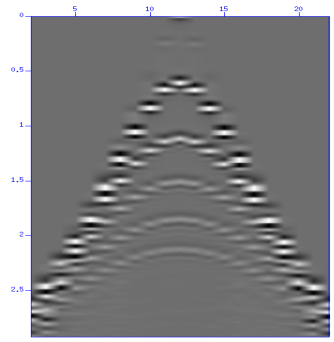
(b) M_1



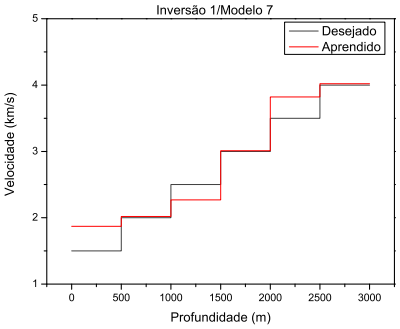
(c) S_5



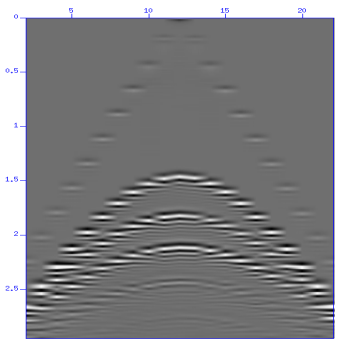
(d) M_5



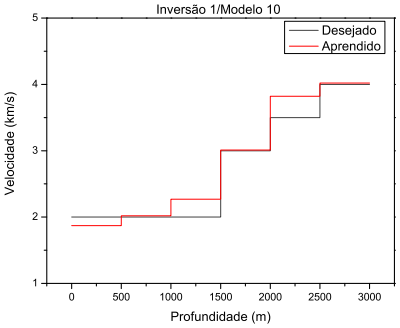
(e) S_7



(f) M_7

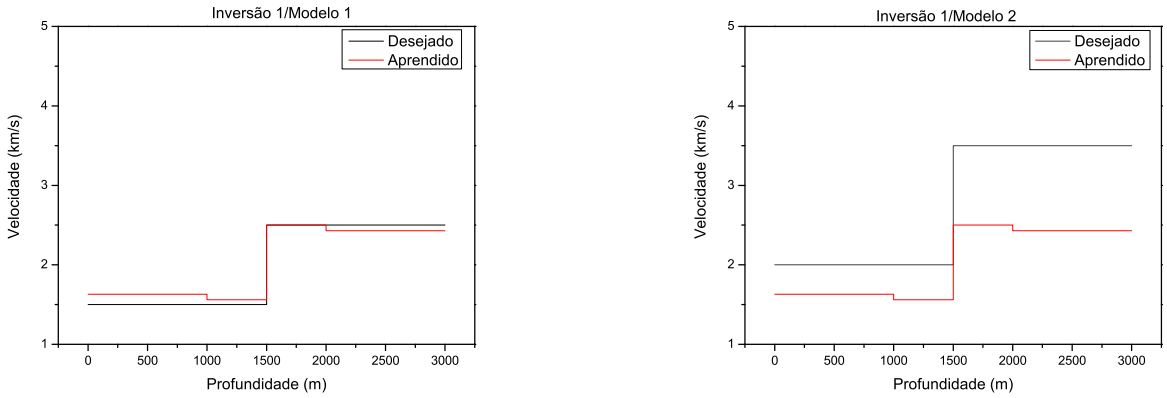


(g) S_{10}



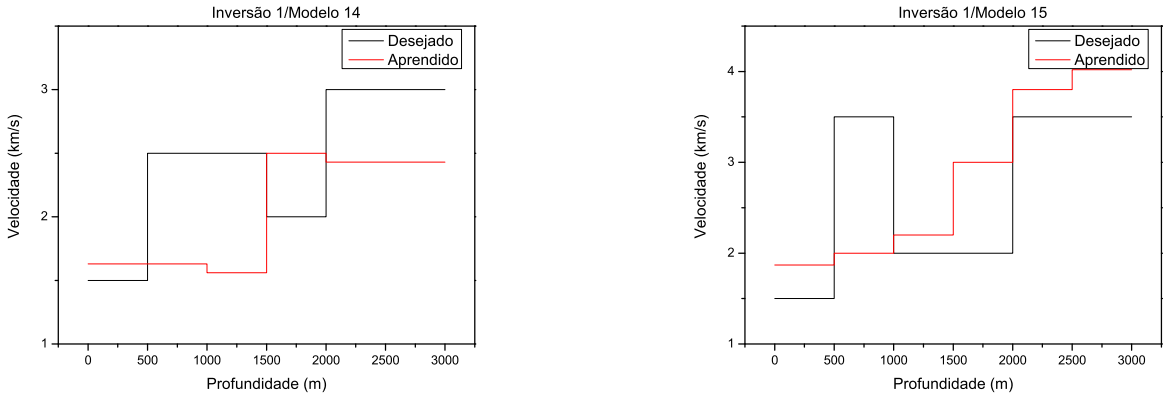
(h) M_{10}

Figura 3.1: À esquerda quatro sismogramas sintéticos e à direita seus modelos desejados correspondentes (linhas pretas) usados para treinar a rede neural. As linhas vermelhas são as saídas da rede neural, com $\eta = 0,3$ e $a = 1,73$.



(a) M_1 (b) M_2

Figura 3.2: Comparação entre as inversões dos modelos M_1 e M_2 realizadas pela rede. Observe que a resposta é a mesma para os dois modelos, tendendo a ajustar-se a M_1.



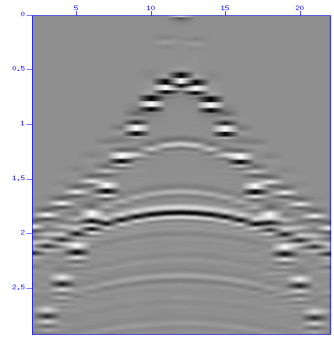
(a) M_{14} (b) M_{15}

Figura 3.3: Resultados da inversão para M_{14} e M_{15}. A rede não inverte bem os modelos de velocidade.

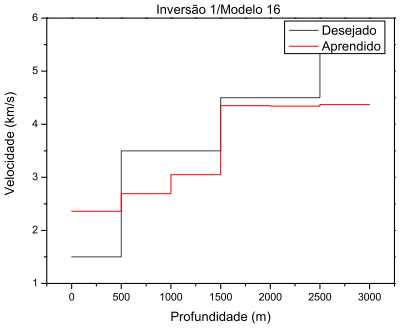
3.2 Capacidade de Generalização da Rede

Vimos na seção anterior que a rede apresentou resultados bons para alguns dos modelos, razoáveis para outros e ruins para um outro grupo. Para exemplos que não fizeram parte da etapa de treinamento, as dificuldades, obviamente, são maiores, mas ainda assim podemos interpretar alguns bons resultados.

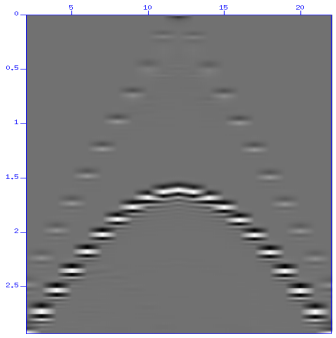
Na Figura 3.4 vemos 4 dos exemplos separados para testar a capacidade de generalização da rede. O ajuste é bastante razoável para os modelos M_{17} e M_{18}, embora, no caso do modelo M_{17}, a exemplo do que é mostrado na Figura 3.2, a rede ajustou as velocidades de M_1 .



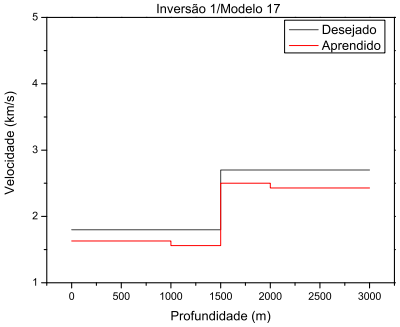
(a) S_{16}



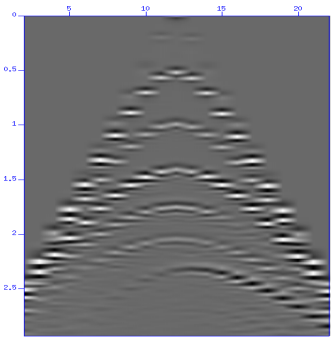
(b) M_{16}



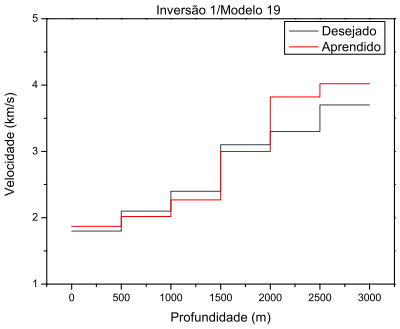
(c) S_{17}



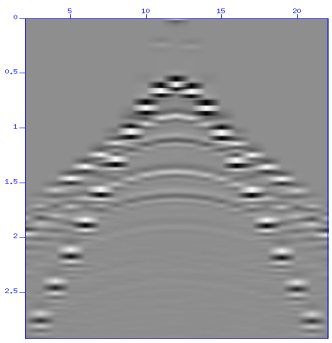
(d) M_{17}



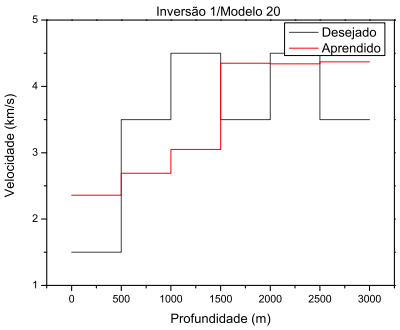
(e) S_{19}



(f) M_{19}



(g) S_{20}



(h) M_{20}

Figura 3.4: Na coluna à esquerda são apresentados quatro sismogramas sintéticos e na da direita seus modelos desejados correspondentes . Estes pares não participaram da etapa de treinamento. As linhas vermelhas são as saídas da rede neural, com $\eta = 0,3$ e $a = 1,73$.

Ainda assim, a despeito do fato de não conseguir ajustar corretamente as velocidades, a rede parece ter uma boa sensibilidade para reconhecer modelos de duas camadas.

Os piores resultados novamente aparecem nos modelos que contêm camadas de baixa velocidade. Esses resultados indicam que a rede tem a tendência de interpolar suas saídas de acordo com os principais modelos do grupo de treinamento. Aparentemente, a rede segue a regra geral de que a velocidade cresce com a profundidade, e falha em reconhecer as exceções, muito provavelmente devido ao número pequeno de exemplos de treinamento. Em nenhum momento a rede conseguiu alcançar velocidades maiores do que $4,5$ km/s, como ilustrado no caso de M_{16} na Figura 3.4. Mais uma vez podemos explicar esse comportamento devido ao fato das velocidades maiores que $4,5$ km/s também serem exceções, aparecendo apenas 4 vezes entre os 90 valores de velocidade existentes no grupo de treinamento.

3.3 Resultados após retirada de alguns exemplos de treinamento

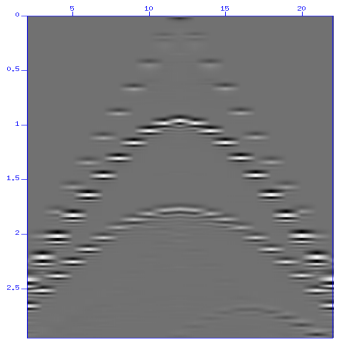
Visando alcançar melhores resultados, efetuamos diversas mudanças no projeto da rede, seja alterando os parâmetros a e η e o critério de parada, ou adicionando/retirando exemplos de treinamento. Em geral, os resultados foram iguais ou piores do que aqueles alcançados com a rede neural original.

Como foi dito na Seção 3.1, a rede foi incapaz de recuperar de forma satisfatória os modelos de 3 camadas e os 5 últimos (M_{11} a M_{15}). Conseguimos um resultado interessante com uma rede de características idênticas a primeira, exceto pela da retirada dos 3 primeiros exemplos do grupo de treinamento. Neste caso a rede atingiu o critério de convergência após 234 iterações. Alguns dos resultados para o grupo de treinamento utilizando essa segunda inversão se encontram na Figura 3.5.

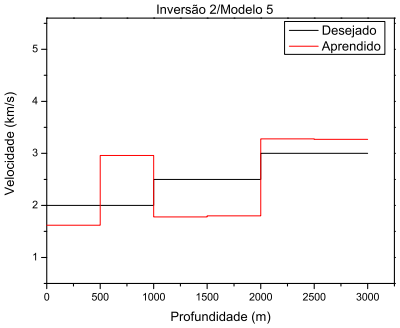
Para essa inversão, os modelos com camadas de baixa velocidade parecem ter tido dominância, pois em todas as respostas aparecem camadas desse tipo. O modelo M_{15} foi o que teve o melhor ajuste.

Embora as respostas aos sismogramas associados a modelos que seguem o padrão comum da natureza (velocidades aumentando com a profundidade) tenham recebido respostas inadequadas, apresentando camadas de baixa velocidade, podemos, ainda assim, encontrar boas respostas para eles da terceira camada em diante. Observando os 3 primeiros exemplos da Figura 3.5, vemos que a rede 'acerta' de certo modo, ou o número de camadas, ou os valores das velocidades em algumas delas.

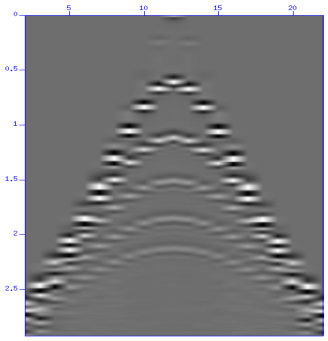
Como já era esperado, a rede perde capacidade de generalização em relação àquela que continha mais exemplos de treinamento. Ainda assim, observamos algumas respostas interessantes, principalmente da terceira camada em diante, assim como ocorreu com os exemplos



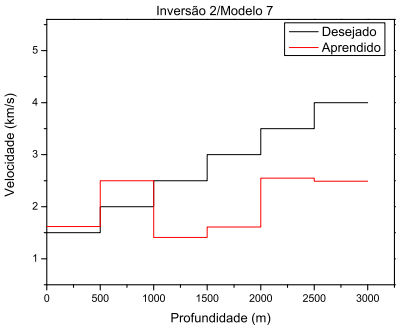
(a) S_5



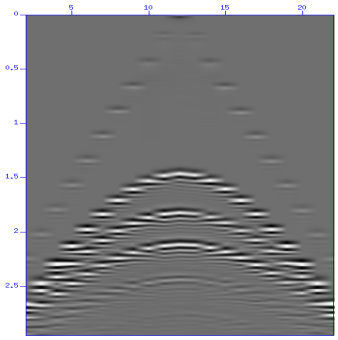
(b) M_5



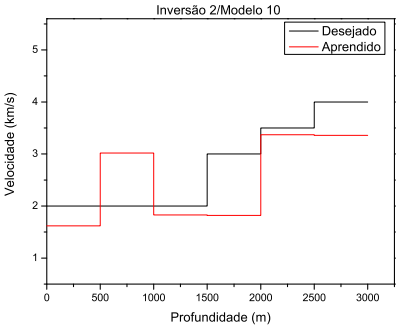
(c) S_7



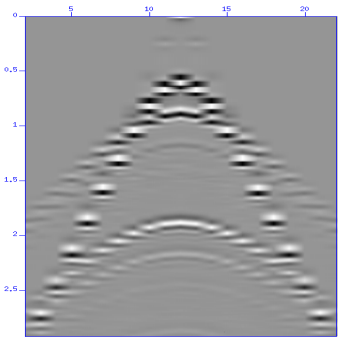
(d) M_7



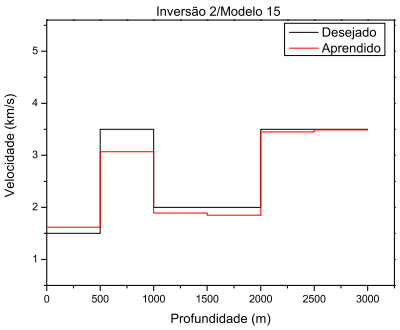
(e) S_{10}



(f) M_{10}



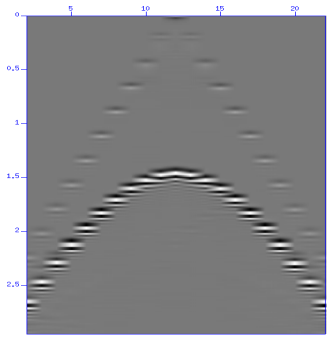
(g) S_{15}



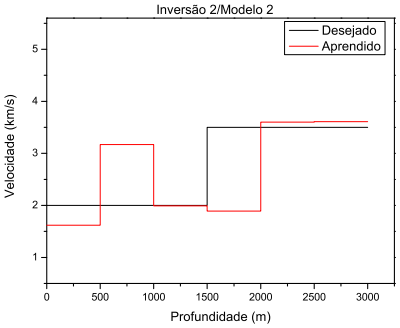
(h) M_{15}

Figura 3.5: À esquerda quatro sismogramas sintéticos utilizados no treinamento, e à direita seus modelos correspondentes (linhas pretas). As linhas vermelhas são as saídas da rede neural. Nesta inversão os modelos M_1, M_2 e M_3 foram retirados do grupo de treinamento.

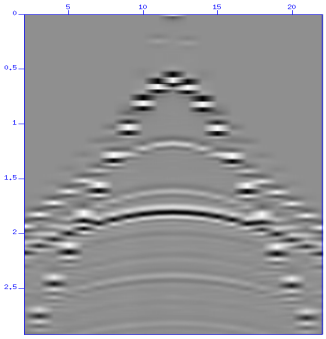
do grupo de treinamento. Na Figura 3.6 apresentamos mais 4 pares sismograma-modelo, desta vez com exemplos que não foram utilizados na etapa de treinamento.



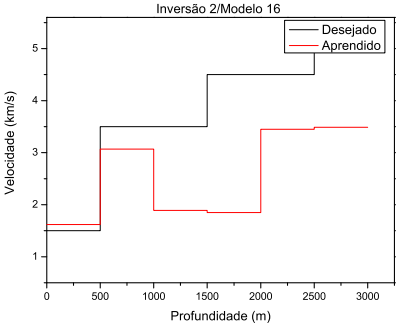
(a) S_2



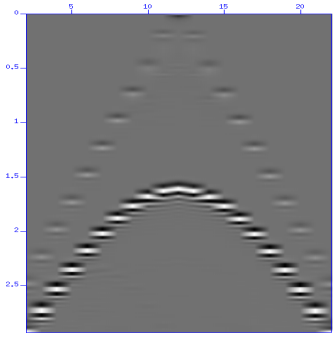
(b) M_2



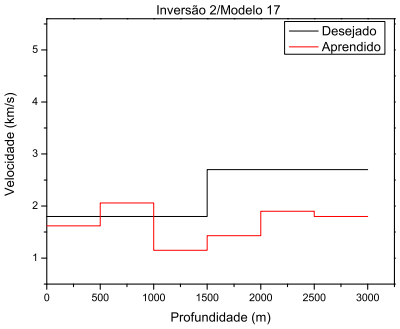
(c) S_{16}



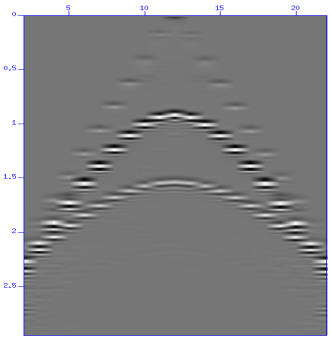
(d) M_{16}



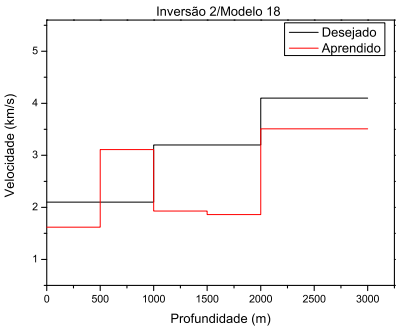
(e) S_{17}



(f) M_{17}



(g) S_{18}



(h) M_{18}

Figura 3.6: Resultados da segunda inversão, para exemplos que não foram utilizados na etapa de treinamento. Note que neste caso M_2 não faz parte dos exemplos de treinamento.

CAPÍTULO 4

Conclusões

Neste trabalho treinamos redes neurais com um número pequeno de amostras de treinamento, visando realizar uma tarefa não-trivial de interpretação de dados sísmicos. A primeira rede neural foi capaz de inverter satisfatoriamente alguns dos modelos do grupo de treinamento, e apresentou resultados coerentes com boa parte dos novos exemplos que lhe foram apresentados. As maiores dificuldades ocorreram nos exemplos onde existiam camadas de baixa velocidade, sendo a rede incapaz de inverter satisfatoriamente os sismogramas.

Fizemos vários testes variando parâmetros e trocando exemplos de treinamento. O único resultado que acrescentou algo aos anteriores foi obtido com uma rede exatamente igual a anterior, mas que foi treinada sem os primeiros 3 exemplos. Esta rede neural obteve resultados interessantes para modelos com camadas de baixa velocidade, e ainda assim conseguiu inverter relativamente bem as 4 últimas camadas, para muitos dos exemplos do grupo de treinamento e alguns dos novos exemplos apresentados.

Consideramos como bons resultados não só os acertos nas velocidades sísmicas das camadas, mas também a capacidade da rede de perceber a tendência geral de mudança de velocidades, bem como o número de camadas. Como a saída da rede é constituída de 6 neurônios independentes, é muito significativo quando a rede reconhece que uma camada tem, por exemplo, 1.000 m, respondendo com dois valores muito próximos para nós subseqüentes. Consideramos esse resultado importante, particularmente, quando a rede é capaz de reconhecer o número de camadas em exemplos que não participaram do treinamento.

O fato de termos conseguido representar um sismograma com 21 traços e 293 amostras em cada um deles, com apenas 45 coeficientes e ainda assim conseguir boas respostas, talvez seja uma das maiores contribuições deste trabalho, pois em nossa pesquisa bibliográfica não encontramos nada semelhante. Acreditamos que a parametrização por série trigonométrica ainda não seja a ideal para o caso dos sismogramas. As pesquisas no campo das *wavelets* apontam para uma utilização promissora deste tipo de parametrização.

Os resultados apresentados neste trabalho são preliminares, e uma investigação mais detalhada ainda precisa ser feita. Acreditamos que sismogramas mais detalhados, um grupo de treinamento maior, e uma parametrização com mais coeficientes permitirão o encontro de melhores resultados.

Agradecimentos

A Deus, pelo dom da vida, e por conceder-me muitas de suas ternas misericórdias.

A Jesus Cristo, pelo exemplo incomparável e por Suas palavras e ações, que me enchem de esperança pelo meu futuro e pelo daqueles que amo.

A meus pais, pela herança moral e genética, que me permite realizar boas coisas na vida.

Em especial à minha mãe, e meu irmão, Allan. Não é possível expressar as experiências que passamos juntos em palavras. Só vocês sabem do que estou falando.

Á minha querida esposa, Lenísia, que de tanto abdicou para que eu pudesse conquistar essa graduação. Por ser a razão principal de meus esforços, por tomar conta de nossa família, e por sempre saber onde guardo minhas coisas.

Aos meus sogros, Antonio e Heleni. Fui abençoado com 2 pais e 2 mães.

Ao meu orientador, Wilson Figueiró, por sua excelente orientação, e por seu exemplo humano.

Aos professores Amin Bassrei e Jacira Freitas, por terem aceitado o convite de participar da banca e por terem sido, cada um a seu modo, excelentes coordenadores do curso.

Aos meus queridos colegas da turma 2007.1. Não vou citar todos, mas não posso deixar de citar Tiago, Cleriston (que não é dessa turma, mas foi adotado), e Rodrigo. É uma honra ter estudado com vocês.

Àqueles que me ajudaram nessa reta final: Luís Eduardo (por me ensinar a por cedilhas em LATEX), Alan (por todo o suporte técnico, instalação de programas, e por me " converter" ao LINUX), Vidal (pela ajuda com vários probleminhas), e Carlos Américo (por encontrar um erro na subrotina que eu nunca teria percebido) e Leonardo Nascimento (the corel man).

Aos meus colegas da UN-FAFEN-BA, pelas trocas de turno que tornaram tudo isso possível.

A todos os funcionários, professores, estudantes, enfim, a todas as pessoas que fazem do curso de Geofísica da UFBA esse grande sucesso.

Referências Bibliográficas

Baum, E., & Haussler, D. (1989) What Size Net Gives Valid Generalization? in Tourektzky, D., Ed., Advances in Neural Information Processing Systems 1, Morgan Kaufmann, 81-90.

Bishop, C. (1995) Neural Networks for Pattern Recognition, Oxford Press.

Cybenco, G. (1989) Approximations by Superpositions of a Sigmoidal Functions: Math of Control, Signals and Systems, 2, 303-314.

German, S., Bienenstock, E., & Doursat, R. (1992) Neural Networks and the Bias/Variance Dilemma, Neural Computation, 4, 1-58.

Haykin, S. (2001) Redes Neurais: Príncipios e Prática; trad. Paulo Martins Engel, Bookman, Porto Alegre.

Hecht-Nielsen, R. (1990) Neurocomputing, Addison-Wesley.

Hornik, K., Stinchcombe, M., & White, H. (1989) Multilayer Feedforward Neural Networks are Universal Approximators, Neural Networks, 2, 359-366.

Kreider, D., Kuller, R.C., Ostberg, D.R. & Perkins. (1972) Introdução à Análise Linear 2, Editora Universal de Brasília, Rio de Janeiro.

Lapedes, A., & Farber, R., 1988, How Neural Networks Work, in Poulton, M.M., American Institute of Physics, 442-456.

Ludwing Jr., O.C., & Montgomery, E. (2007) Redes Neurais: Fundamentos e Aplicações com Programas em C, Editora Ciência Moderna, Rio de Janeiro.

Poulton, M.M. (2003), Multi-Layer Perceptrons and Back-Propagation Learning, Neural Networks and Geophysics, chapter 3, Elsevier.

Röth, G., & Tarantola, A. (1991) Use of neural networks for the inversion of seismic data, SEG-Expanded Abstracts, vol. 1, pp. 302-305, SEG Publications, Tulsa.

Röth, G., & Tarantola, A. (1994) Neural networks and inversion of seismic data, Journal of Geophysical Research, vol.99, no. B4, pp.6753-6768.

Santos, R.H.M., & Figueiró, W.M. (2006) Modelagem Acústica Bidimensional Usando Diferentes Parametrizações de Campos de Velocidades, Revista Brasileira de Geofísica, vol. 24, número 1, pp. 103-115.

Widrow, B., & Hoff, M. (1960) Adaptive Switching Circuits, Convention Record, 96-104.

APÊNDICE A

Calculando os Coeficientes da Série Trigonométrica

Um sismograma pode ser visto como uma função $A(x,t)$, e também como uma matriz, A, com um número p de colunas (traços) e com q linhas (amostras temporais por traço), tendo como entrada na posição m, n a amplitude $A_{m,n}$ do m-ésimo traço em sua n-ésima amostra temporal. Sendo assim, A possui a seguinte forma:

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} & A_{1,3} & \dots & A_{1,m} & \dots & A_{1,p} \\ A_{2,1} & A_{2,2} & A_{2,3} & \dots & A_{2,m} & \dots & A_{2,p} \\ A_{3,1} & A_{3,2} & A_{33} & \dots & A_{3,m} & \dots & A_{3,p} \\ \vdots & \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ A_{n,1} & A_{n,2} & A_{n,3} & \dots & A_{n,m} & \dots & A_{n,p} \\ \vdots & \vdots & \vdots & \dots & \ddots & \dots & \ddots \\ A_{q,1} & A_{q,2} & A_{q,3} & \dots & A_{q,m} & \dots & A_{q,p} \end{bmatrix} \tag{A.1}$$

Consideremos a fórmula (2.1) que fornece $\alpha_{i,j}$, na qual $f_i(x)$ e $g_j(t)$ são dadas pelas Eqs. (1.4), (1.5) e (1.6); e $D = [0, l] \times [0, \tau]$; sendo que na Eq. (1.5) z é substituído por t e Z por $T{=}\dfrac{\pi(2t-\tau)}{\tau}$, onde τ é o tempo máximo de amostragem.

Se B é o denominador da Eq. (2.1), existem quatro casos para o seu cálculo, a saber:

- (i) i é par e j é par,
- (ii) i é par e j é impar,
- (iii) i é impar e j é par e
- (iv) i é impar e j é impar

Para o caso (i) temos:

$$f_i(x) = \frac{1}{2}\left\{2\cos\left(\frac{iX}{2}\right)\right\} = \cos\left(\frac{iX}{2}\right), \tag{A.2}$$

$$g_j(t) = \frac{1}{2}\left\{2\cos\left(\frac{jT}{2}\right)\right\} = \cos\left(\frac{jT}{2}\right), \tag{A.3}$$

$$B = \int_0^{\tau}\int_0^{l} \cos^2\left(\frac{iX}{2}\right)\cos^2\left(\frac{jT}{2}\right)dxdt, \tag{A.4}$$

ou

$$B = \int_0^{l} \cos^2\left(\frac{iX}{2}\right)dx \int_0^{\tau}\cos^2\left(\frac{jT}{2}\right)dt. \tag{A.5}$$

Segue abaixo a resolução das duas integrais que aparecem em (A.5):

$$\int_0^{l} \cos^2\left(\frac{iX}{2}\right)dx = \int_0^{l}\cos^2\left[\frac{i\pi(2x-l)}{2l}\right]dx =$$

$$= \int_0^{l}\frac{1}{2}\left\{\cos\left[\frac{i\pi(2x-l)}{l}\right]+1\right\}dx = \frac{1}{2}\int_0^{l}\cos\left[\frac{i\pi(2x-l)}{l}\right]dx + \frac{l}{2} = \frac{l}{2} \tag{A.6}$$

e

$$\int_0^{\tau}\cos^2\left(\frac{jT}{2}\right)dt = \int_0^{\tau}\cos^2\left[\frac{j\pi(2t-\tau)}{2\tau}\right]dt =$$

$$= \int_0^{\tau}\frac{1}{2}\left\{\cos\left[\frac{j\pi(2t-\tau)}{\tau}\right]+1\right\}dt = \frac{1}{2}\int_0^{\tau}\cos\left[\frac{j\pi(2t-\tau)}{\tau}\right]dt + \frac{\tau}{2} = \frac{\tau}{2} \tag{A.7}$$

Portanto para o caso (i), $B = l\tau/4$.

Vejamos agora o que acontece no caso (ii):

$$f_i(x) = \cos\left(\frac{iX}{2}\right), \tag{A.8}$$

$$g_j(t) = \frac{1}{2}\left\{2\,\mathrm{sen}\left[\frac{(j+1)T}{2}\right]\right\} = \mathrm{sen}\left[\frac{(j+1)T}{2}\right], \tag{A.9}$$

e

$$B = \int_0^{\tau}\int_0^{l}\cos^2\left(\frac{iX}{2}\right)\mathrm{sen}^2\left[\frac{(j+1)T}{2}\right]dxdt =$$

$$B = \int_0^{l}\cos^2\left(\frac{iX}{2}\right)dx\int_0^{\tau}\mathrm{sen}^2\left[\frac{(j+1)T}{2}\right]dt \tag{A.10}$$

Segue abaixo a resolução da segunda integral de A.10:

$$\int_0^{\tau} \operatorname{sen}^2\left[\frac{(j+1)T}{2}\right] dt = \int_0^{\tau} \operatorname{sen}^2\left[\frac{(j+1)\pi(2t-\tau)}{2\tau}\right] dt =$$

$$= \frac{1}{2}\int_0^{\tau}\left\{1-\cos\left[\frac{(j+1)\pi(2t-\tau)}{\tau}\right]\right\} dt = \int_0^{\tau}\frac{1}{2}dt - \frac{1}{2}\int_0^{\tau}\cos\left[\frac{(j+1)\pi(2t-\tau)}{\tau}\right] dt =$$

$$= \frac{\tau}{2} - \frac{1}{2}\operatorname{sen}\left[\frac{(j+1)\pi(2t-\tau)}{\tau}\right] = \frac{\tau}{(j+1)2\pi}\Bigg|_0^{\tau} = \frac{\tau}{2}. \tag{A.11}$$

Novamente $B = l\tau/4$. Usando o mesmo raciocínio o leitor pode conferir os casos restantes e verificar que B sempre será igual a $l\tau/4$. Portanto, podemos reescrever (2.1) da seguinte forma:

$$\alpha_{i,j} = \frac{4}{l\tau}\iint_D A(x,t) f_i(x) g_j(t) dD. \tag{A.12}$$

Não conhecemos uma expressão contínua para a função $A(x,t)$, porém a conhecemos por diversos pontor $(x_i,\, t_j)$, que são as amplitudes registradas nos sismogramas no traço i e no instante de tempo t_j. Discretizando-se o sismograma conforme matriz apresentada em (A.1), e considerando-se que a amplitude se mantém constante em cada célula de discretização, temos:

$$\alpha_{i,j} \cong \frac{4}{l\tau}\sum_{m=1}^{p}\sum_{n=1}^{q} A_{mn}\int_{\frac{(m-1)l}{p}}^{\frac{ml}{p}}\int_{\frac{(n-1)\tau}{q}}^{\frac{n\tau}{q}} f_i(x) g_j(t) dx dt =$$

$$= \frac{4}{l\tau}\sum_{m=1}^{p}\sum_{n=1}^{q} A_{mn}\int_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} f_i(x) dx \int_{\frac{(n-1)\tau}{q}}^{\frac{n\tau}{q}} g_j(t) dt. \tag{A.13}$$

Daí, obtêm-se a Eq. (2.2).

A seguir, são apresentada as demonstrações das fórmulas matemáticas de K_0, M_0, K_i, L_i, M_j, e N_j; apresentadas nas Eqs. (2.3), (2.4), (2.5), (2.6), (2.7) e (2.8); respectivamente. Portanto, partindo-se das integrais que comparecem no último membro da Eq. (A.13), seguem as demonstrações anteriormente mencionadas.

Cálculo de K_i

Como i é par, temos que $f_i(x) = \cos\left[\frac{iX}{2}\right]$, logo:

$$\int_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} f_i(x)dx = \int_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} \cos\left[\frac{i\pi(2x-l)}{2l}\right] dx = \operatorname{sen}\left[\frac{i\pi(2x-l)}{2l}\right] \frac{2l}{2\pi i}\Bigg|_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} =$$

$$= \frac{l}{i\pi}\left\{\operatorname{sen}\left[\frac{i\pi}{2p}(2m-p)\right] - \operatorname{sen}\left[\frac{i\pi}{2p}\left(2(m-1)-p\right)\right]\right\}. \quad \text{(A.14)}$$

O que implica em:

$$K_i = \frac{1}{i}\left\{\operatorname{sen}\left[\frac{i\pi}{2p}(2m-p)\right] - \operatorname{sen}\left[\frac{i\pi}{2p}\left(2(m-1)-p\right)\right]\right\} \quad \text{(A.15)}$$

Os últimos termos de (A.14) e (A.15) se distinguem pelo fator $\frac{l}{\pi}$, que quando colocada para fora dos somatórios de (A.13) faz com que o fator $4/l\mathcal{T}$ se transforme em $4/\pi\mathcal{T}$.

Cálculo de L_i

Como i é ímpar, temos que $f_i(x) = \operatorname{sen}\left[\frac{(i+1)X}{2}\right]$, logo:

$$\begin{aligned}\int_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} f_i(x)dx &= \int_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} \operatorname{sen}\left[\frac{(i+1)\pi(2x-l)}{2l}\right] dx = -\cos\left[\frac{(i+1)\pi(2x-l)}{2l}\right] \frac{l}{\pi(i+1)}\Bigg|_{\frac{(m-1)l}{p}}^{\frac{ml}{p}} = \\ &= \frac{l}{(i+1)\pi}\left\{\cos\left[\frac{(i+1)\pi(2(m-1)-p)}{2p}\right] - \cos\left[\frac{(i+1)(2m-p)\pi}{2p}\ )\right]\right\}. \quad \text{(A.16)}\end{aligned}$$

O que implica em:

$$L_i = \frac{1}{1+i}\left\{\cos\left[\frac{(i+1)\pi}{2p}(2(m-1)-p)\right] - \cos\left[\frac{(i+1)\pi}{2p}(2m-p)\right]\right\}. \quad \text{(A.17)}$$

Cálculo de M_j

Como j é par, temos que $g_j(t) = \cos\left[\frac{jT}{2}\right]$, logo:

$$\begin{aligned}\int_{\frac{(n-1)\tau}{q}}^{\frac{n\tau}{\tau}} g_j(t)dt &= \int_{\frac{(n-1)\tau}{q}}^{\frac{n\tau}{q}} \cos\left[\frac{j\pi(2t-\tau)}{2\tau}\right] dt = \\ &= \frac{\tau}{j\pi}\left\{\operatorname{sen}\left[\frac{j\pi(2n-q)}{2q}\right] - \operatorname{sen}\left[\frac{j(2(n-1)-q)\pi}{2q}\ )\right]\right\}. \quad \text{(A.18)}\end{aligned}$$

o que implica em:

$$M_j = \frac{1}{j}\left\{\text{sen}\left[\frac{j\pi}{2q}(2n-q)\right] - \text{sen}\left[\frac{j\pi}{2q}\left(2(n-1)-q\right)\right]\right\}. \tag{A.19}$$

Cálculo de N_j

Como j é ímpar, temos que $g_j(t) = \text{sen}\left[\frac{(j+1)T}{2}\right]$, logo:

$$\int_{\frac{(n-1)\tau}{q}}^{\frac{n\tau}{\tau}} g_j(t)dt = \int_{\frac{(n-1)\tau}{q}}^{\frac{n\tau}{q}} \text{sen}\left[\frac{(j+1)\pi(2t-\tau)}{2\tau}\right]dt =$$

$$= \frac{\tau}{(j+1)\pi}\left\{\cos\left[\frac{(j+1)\pi(2(n-1)-q)}{2q}\right] - \cos\left[\frac{(j+1)(2n-q)\pi}{2q}\ )\right]\right\}. \tag{A.20}$$

O que implica em:

$$N_j = \frac{1}{1+j}\left\{\cos\left[\frac{(j+1)\pi}{2q}(2(n-1)-q)\right] - \cos\left[\frac{(j+1)\pi}{2q}(2n-q)\right]\right\}. \tag{A.21}$$

Cálculo de K_0

Em (A.15), podemos observar que para $i = 0$, K_i não é definido. Precisamos utilizar alguns artifícios matemáticos para obter a expressão de K_0.

Inicialmente multiplicamos e dividimos o primeiro termo de (A.15) por $\pi(2m-p)/2p$, e o segundo por $\pi(2(m-1)-p)/2p$, chegando na seguinte expressão:

$$K_i = \frac{\pi(2m-p)}{2p}\left[\frac{\text{sen}\left(\frac{i\pi(2m-p)}{2p}\right)}{\frac{i\pi(2m-p)}{2p}}\right] - \frac{\pi(2(m-1)-p)}{2p}\left[\frac{\text{sen}\left(\frac{i\pi(2(m-1)-p)}{2p}\right)}{\frac{i\pi(2(m-1)-p)}{2p}}\right]. \tag{A.22}$$

Quando i tende a zero, as expressões entre colchetes em (A.22) tendem a 1, de acordo com o limite fundamental trigonométrico:

$$\lim_{x\to 0}\frac{\text{sen}\,x}{x} = 1. \tag{A.23}$$

Desse modo, (A.22) se reduz a:

$$K_0 = \frac{\pi}{p}. \tag{A.24}$$

Cálculo de M_0

Utilizando-se os mesmos argumentos para a obtenção de K_0, temos:

$$M_0 = \frac{\pi}{q}. \tag{A.25}$$