Curso de Redes Neurais utilizando o MATLAB

Victoria Yukie Matsunaga

Belém-Pará-Brasil
2012

Esta apostila tem como principal objetivo fornecer um material de auxílio ao Curso de Redes Neurais utilizando o MATLAB. Inicialmente, alguns conceitos básicos necessários para o entendimento desta ferramenta serão abordados e, posteriormente, alguns comandos e funções do software serão apresentados. A elaboração deste material foi baseada principalmente nas seguintes bibliografias:

- Guia do Usuário do MATLAB.
- *Neural Networks: a comprehensive foundation*, S. Haykin, New York: MacMillan College Publishing Co., 2nd Edition.

O Guia do Usuário do MATLAB encontra-se disponível no site da The MathWorks, Inc: `http://www.mathworks.com/`. As demais fontes estão listadas no final deste material.

Sumário

Capítulo 1

Redes Neurais Artificiais

As redes neurais artificiais (RNAs) são sistemas de computação adaptativos inspirados nas características de processamento de informação encontradas nos neurônios reais e nas características de suas interconexões [1].

O estudo de RNAs é baseado no comportamento das redes neurais biológicas. O cérebro humano possui uma enorme capacidade de processar informações, como por exemplo, o reconhecimento de fala e a segmentação de imagens. E, a partir desta análise, pesquisas são realizadas com a finalidade de reproduzir as reações do cérebro em máquinas (rede neural), tendo como resultado bastante promissor (apesar do pouco conhecimento do cérebro humano).

As RNAs são formadas por um conjunto de neurônios artificiais que interagem entre si, semelhantes ao funcionamento dos neurônios biológicos. Basicamente, são sistemas paralelos de computação e suas implementações podem ser em hardware (realiza uma determinada tarefa a partir de componentes eletrônicos) ou em software (simulações por programação em um computador digital).

As redes neurais têm a capacidade de coletar, utilizar e armazenar informações baseadas em experimentos (aprendizagem). O processo de aprendizagem é feito a partir de algoritmos de aprendizagem, onde os pesos sinápticos da rede são modificados de uma forma ordenada para alcançar o resultado desejado.

A partir da informação contida em pares de entradas e saídas, contidos em um banco de dados, denotados por $(\bar{p_1}, \bar{t_1}), (\bar{p_2}, \bar{t_2}), \ldots, (\bar{p_Q}, \bar{t_Q})$ (as chamadas amostras de treinamento), as redes neurais devem "aprender" e generalizar o conhecimento contido nas amostras, de forma a responder de forma adequada quando entradas não presentes no conjunto de treinamento forem apresentadas à rede.

Isto é possível devido ao fato de que a formulação é baseada numa representação (simplificada) de uma rede real de neurônios. Dessa forma, problemas relacionados a reconhecimento de padrões podem ser solucionados através da implementação e uso adequados desta técnica.

Algumas características importantes em uma rede neural artificial são:

- Robustez e tolerância a falhas: a eliminação de alguns neurônios não afeta substancialmente o seu desempenho global.
- Flexibilidade: pode ser ajustada a novos ambientes por meio de um processo de aprendizagem, sendo capaz de aprender novas ações com base na informação contida nos dados de treinamento.
- Processamento de informação incerta: mesmo que a informação fornecida esteja incompleta, afetada por ruído, ainda é possível obter-se um raciocínio correto.
- Paralelismo: um imenso número de neurônios está ativo ao mesmo tempo. Não existe a restrição de um processador que obrigatoriamente trabalhe uma instrução após outra.

A estrutura de uma rede neural é paralelamente distribuída, e possui habilidade de aprender. Portanto, sua utilização oferece alguns benefícios:

- Capacidade de realizar mapeamentos não-lineares entre entrada e saída;
- Adaptabilidade, mediante a um algoritmo de aprendizado;

- Tolerância à falhas
- Uniformidade de análise e projeto
- Analogia neurobiológica

1.1 Neurônios Biológicos

O esforço para entender o cérebro se tornou mais fácil pelo trabalho pioneiro de Ramón y Cajál (1911) [1], que introduziu a ideia dos neurônios como constituintes estruturais do cérebro. Tipicamente, os neurônios biológicos são de cinco a seis ordens de grandeza mais lentas que as portas lógicas em silício; as quais atuam na faixa de nanossegundos (10^{-9}s).

Entretanto, o cérebro pode compensar a taxa de operação relativamente lenta de um neurônio pelo número muito elevado de neurônios (células nervosas), com conexões maciças entre si.

Estima-se que haja aproximadamente 10 bilhões de neurônios no córtex humano e 60 trilhões de sinapses ou conexões [2]. O cérebro humano, representado pela rede neural (nervosa), é constituído por estruturas, neurônios, que interagem entre si, armazenando e tomando decisões adequadas a partir das informações recebidas.

Um neurônio (Figura 1.1 [3]) é composto por um corpo celular (ou soma), um axônio tubular e várias ramificações arbóreas conhecidas como dendritos. Os dendritos (zonas receptivas) formam uma malha de filamentos finíssimos ao redor do neurônio. Ao passo que o axônio (linhas de transmissão) consta de um tubo longo e fino que ao final se divide em ramos que terminam em pequenos bulbos que quase tocam os dendritos dos outros neurônios. O pequeno espaço entre o fim do bulbo e o dendrito é conhecido como sinapse, que tem como papel fundamental a memorização da informação.

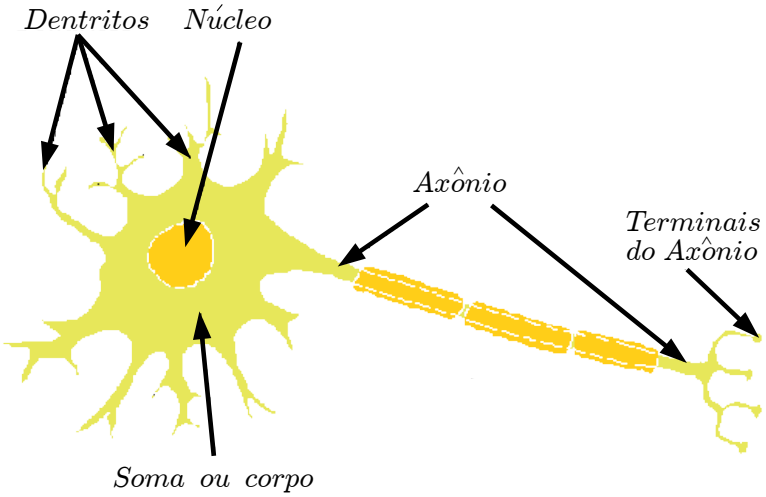

Figura 1.1: Modelo de um neurônio biológico.

1.2 Modelo Artificial do Neurônio

Basicamente, todos os tipos de redes neurais apresentam a mesma unidade de processamento: um neurônio artificial, que simula o comportamento do neurônio biológico. Esse neurônio artificial possui várias entradas, que correspondem às conexões sinápticas com outras unidades similares a ele, e uma saída, cujo valor depende diretamente da somatória ponderada de todas as saídas dos outros neurônios a esse conectado.

O modelo artificial de neurônio é mostrado na Figura 1.2, sendo uma generalização do modelo de McCulloch e Pitts [4, 1]. Esse modelo inclui um sinal adicional *bias* (b) que favorece ou limita a possibilidade de ativação do neurônio. O processo sináptico é representado pelos pesos (ω) que amplificam cada um dos sinais recebidos. A chamada função de ativação (f) modela a forma como o neurônio responde ao nível de excitação, limitando e definindo a saída da rede neural.

A função de ativação pode ter diferentes representações. Os três tipos básicos de função de ativação são: limiar, linear e sigmóide. A escolha do tipo varia de acordo com

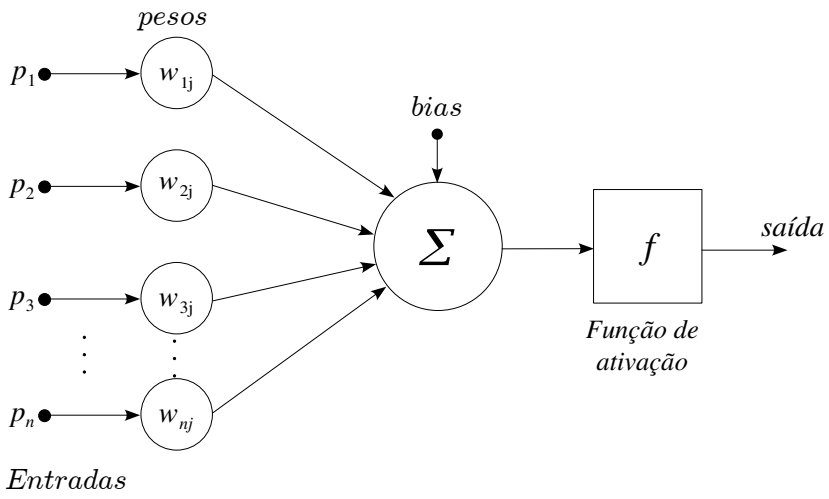

Figura 1.2: Modelo artificial de neurônio biológico.

o objetivo do projeto [1].

1.3 Arquitetura da Rede Neural

As disposições (arquitetura) de uma rede neural são de diversas formas, conforme a necessidade da aplicação, estando diretamente ligadas com o algoritmo de aprendizagem usado para treinar a rede. Basicamente, os itens que compõem a estrutura de uma rede neural são, portanto, sujeito a modificações, sendo os seguintes [1]:

- Camadas intermediárias
- Quantidade de neurônios
- Função de transferência
- Algoritmo de aprendizado

A maneira pela qual os neurônios de uma rede neural estão estruturados está intimamente ligada ao algoritmo de aprendizagem a ser usado para treinar a rede. Pode-se,

portanto, falar de algoritmos (regras) de aprendizagem utilizados no projeto de redes neurais como sendo *estruturados*. Em geral, podemos classificar três tipos de arquiteturas de rede fundamentalmente diferentes, como descritos a seguir.

1.3.1 Redes Alimentadas Adiante com Camada Única

Em uma rede neural em camadas, os neurônios estão organizados na forma de camadas (Figura 1.3). O termo "camada única" se refere à camada de saída de nós computacionais (neurônios).

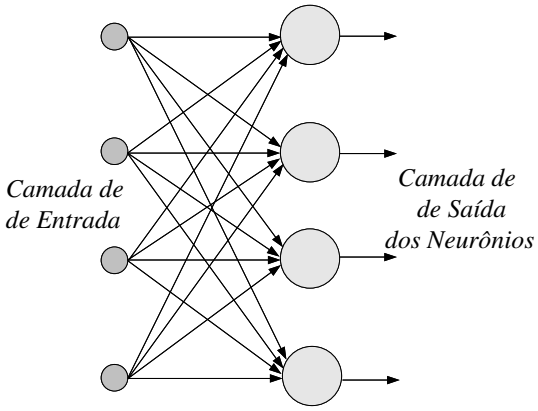

Figura 1.3: Redes alimentadas adiante com camada única.

1.3.2 Redes Alimentadas Diretamente com Múltiplas Camadas

Esse tipo de arquitetura se distingue pela presença de uma ou mais camadas ocultas (ou intermediárias), cujos nós computacionais são chamados de neurônios ocultos ou unidades ocultas. A função dos neurônios ocultos é intervir entre a entrada externa e a saída da rede de uma maneira útil. Adicionando-se uma ou mais camadas ocultas, tornamos a rede capaz de extrair estatísticas de ordem elevada.

A Figura 1.4 mostra um exemplo de uma RNA de 2 camadas com 4 entradas e 2 saídas.

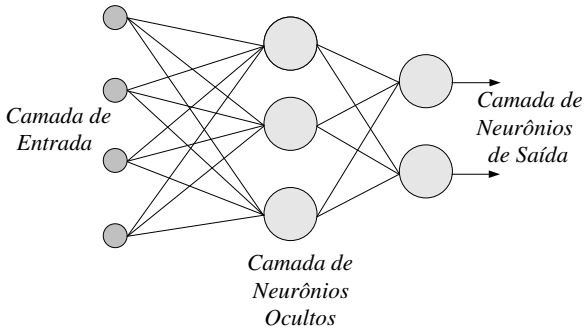

Figura 1.4: Redes alimentadas diretamente com múltiplas camadas.

1.3.3 Redes Recorrentes

Elas se distinguem das redes alimentadas adiante por terem pelo menos um laço de realimentação, com a presença ou não de neurônios ocultos. A presença de laços de realimentação tem um impacto profundo na capacidade de aprendizagem da rede e no seu desempenho.

1.4 Aprendizado

O aprendizado das redes neurais pode ser na forma supervisionada ou não supervisionada.

No aprendizado supervisionado, o instrutor confere o quanto a rede está próxima de uma solução aceitável, adaptando na concepção do treinamento os pesos entre os neurônios, de modo a prover uma menor diferença entre as saídas desejadas.

No aprendizado não supervisionado, inicialmente, as saídas da rede não são conhecidas, funcionando de modo a distinguir classes de padrões diferentes dos dados apresentados à rede, através de algoritmos de aprendizado baseados geralmente em conceitos de vizinhança e agrupamento. A rede é ajustada de acordo com regularidades estatísticas dos dados de entrada, de tal forma que ela cria categorias, otimizando em relação aos

parâmetros livres da rede uma medida da quantidade que é independente da tarefa a ser executada.

1.5 Regra de Aprendizado por Retropropagação (*Back-propagation*)

O algoritmo *Backpropagation* procura achar iterativamente a mínima diferença entre as saídas desejadas e as saídas obtidas pela rede neural, com o mínimo de erro. Dessa forma, ajustando os pesos entre as camadas através da retropropagação do erro encontrado em cada iteração [5, 1].

Essa regra é um dos tipos de treinamento supervisionado, onde a rede é analisada em dois casos: na sua propagação (camada por camada) e principalmente, na sua retropropagação (análise contrária à propagação), *Backpropagation*. No primeiro, os pesos sinápticos ω_{ji} (peso sináptico conectando a saída do neurônio i à entrada do neurônio j) da rede são todos fixos. No segundo, os pesos são todos ajustados.

Basicamente, um padrão de entrada é aplicado como um estímulo aos elementos da primeira camada da rede, que é propagado por cada uma das outras camadas até que a saída seja gerada a. Ela é então comparada com a saída desejada a_d (gerando um sinal de erro e para cada elemento de saída). O sinal de erro é então retropropagado da camada de saída para cada elemento da camada intermediária anterior que contribui diretamente para a formação da saída.

Entretanto, cada elemento da camada intermediária recebe apenas uma porção do sinal de erro total, proporcional apenas à contribuição relativa de cada elemento na formação da saída original. Este processo se repete, camada por camada, até que cada elemento da rede receba um sinal de erro que descreva sua contribuição relativa para o erro total.

Com base no sinal de erro recebido, os pesos sinápticos são então atualizados (de acordo com uma regra de correção de erro) para cada elemento de modo a fazer a rede

convergir para o valor de saída desejada a_d. A ilustração do algoritmo *Backpropagation* pode ser verificada na Figura 1.5.

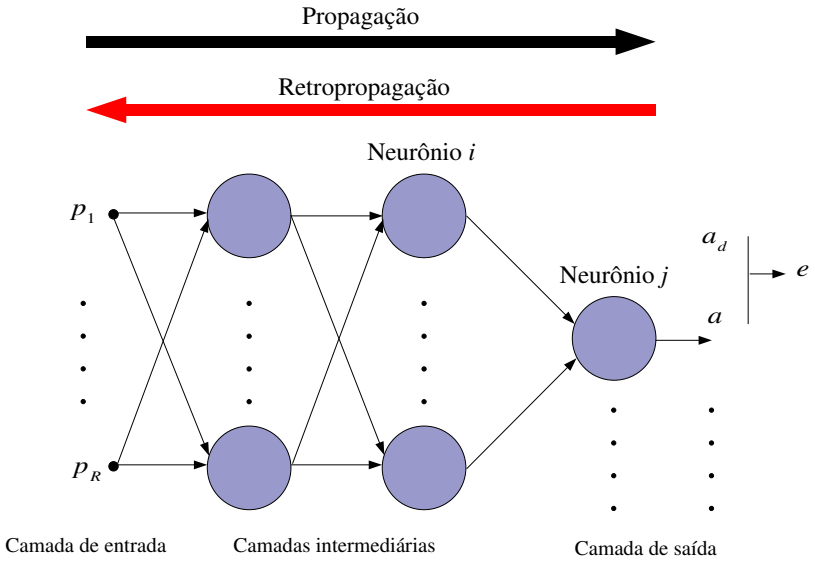

Figura 1.5: Ilustração do algoritmo *Backpropagation*.

Capítulo 2

MATLAB

O MATLAB (*Matrix Laboratory*), software desenvolvido pela *The MathWorks, Inc*, é um sistema interativo e uma linguagem de programação baseados numa matriz, onde as soluções e problemas são expressos em linguagem matemática. Através da utilização do MATLAB é possível realizar análise numérica, operações com matrizes, construção de gráficos e criação de interfaces com o usuário.

Uma das principais vantagens do software é a facilidade de escrever e depurar um programa, se comparado a outras linguagens de programação (tais como C, Basic, Pascal ou Fortran). Além disso, possui diversas funções matemáticas, matriciais e gráficas que simplificam e minimizam a estrutura do programa.

O MATLAB também dispõe de diversas bibliotecas ou ferramentas (*toolboxes*) para aplicações específicas, como por exemplo, Redes Neurais, Lógica Fuzzy, Otimização de Sistemas, *Wavelets*, Cálculo Simbólico, Processamento de Sinais e outras. O software possui versões para diferentes ambientes operacionais: Windows, Linux, UNIX, Macintosh.

O *toolbox* abordado neste curso será o de **Redes Neurais**, onde a versão do MATLAB é para ambiente Windows.

A utilização do *toolbox* de redes neurais no MATLAB pode ser através de interface gráfica (NNTool), linhas de comando ou arquivo ".m". Nas subsequentes seções serão descritas esses modos de utilização.

2.1 Interface Gráfica NNTool

A interface gráfica NNTool (*Neural Network Toolbox*) permite importar, criar, utilizar e exportar dados de redes neurais. O acesso ao NNTool é realizado digitando **nntool** na janela de comandos do MATLAB. Uma janela denominada *Network/ Data Manager* será aberta, conforme a Figura 2.1

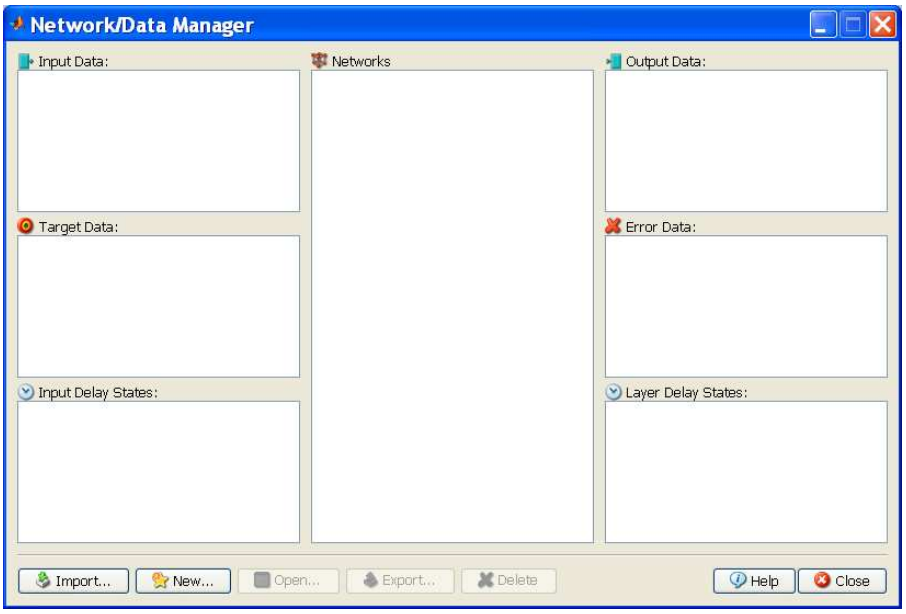

Figura 2.1: NNTool.

Os padrões (dados de entrada e de saída) e a arquitetura da rede são definidos de acordo com a Figura 2.2

É possível visualizar a rede neural pressionando o botão *View* como mostra a Figura 2.3.

Após a criação da rede, passamos para a fase de treinamento (Figura 2.4). Nesta etapa, determinamos cada parâmetro de treinamento, tais como dados de validação e teste, número de épocas, valores dos pesos ou bias, etc. Para iniciar o processo de treinamento pressiona-se o botão *Train Network*.

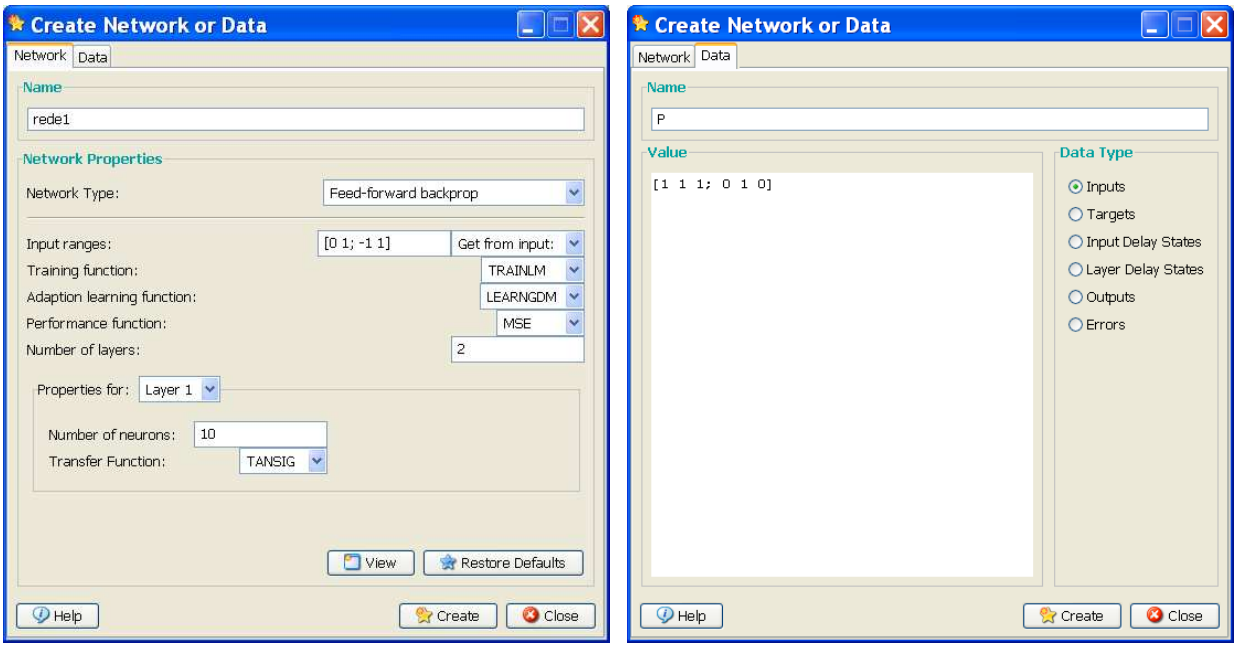

Figura 2.2: Definição dos padrões e das características da rede.

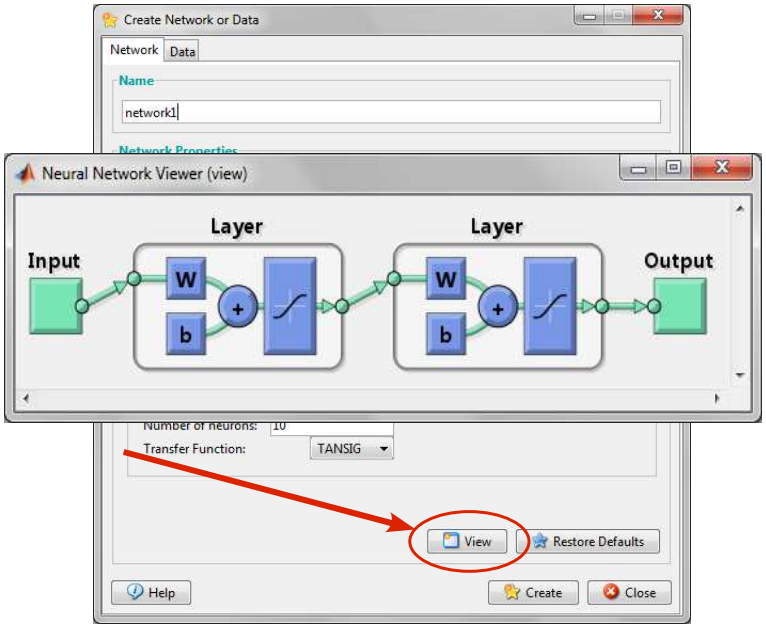

Figura 2.3: Visualização da rede.

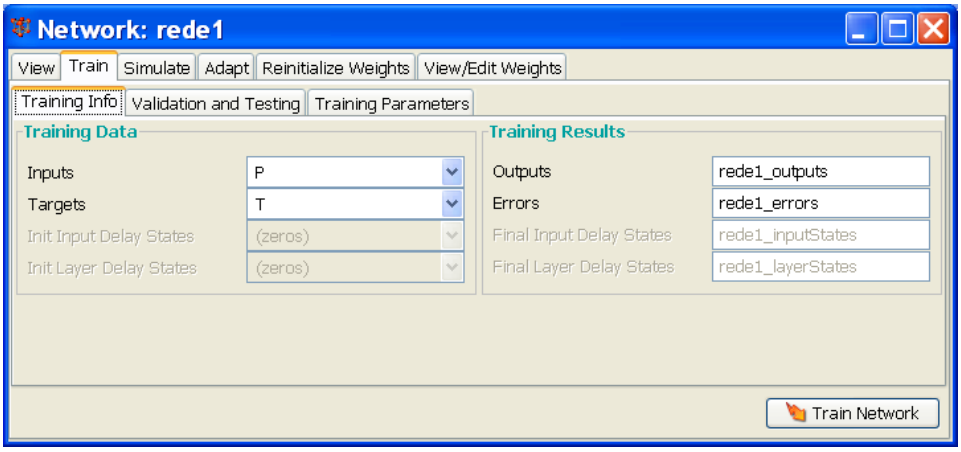

Figura 2.4: Visualização da rede.

2.2 Linhas de Comando e arquivos ".m" para criação de uma rede neural

Uma rede neural também pode ser desenvolvida através de comandos escritos na Janela de Comandos ou no ambiente de edição do MATLAB. Para o entendimento desta seção, utilizaremos uma rede do tipo *feed-forward* (sem realimentação) como exemplo.

As etapas para a implementação de uma rede são:

1. Definição dos dados de entrada e saída

2. Inicialização da rede

   A função **newff** cria uma rede do tipo *Backpropagation*.

```
net = newff (  [min(P) max(P)],          (limites dos padrões de entrada - mín. e máx.)
               [10 10 1],                (número de neurônios de cada camada)
               'purelin','tansig', 'tansig',   (função de transferência de cada camada)
               'traingd';                (algoritmo de treinamento)
```

   Os tipos de funções de transferência e de algoritmos de aprendizado são descritos nas Tabelas 2.1 e 2.2 .

Tabela 2.1: Funções de transferência.

Função	Descrição
purelin	Linear
tansig	Tangente hiperbólica
logsig	Sigmóide
satlin	Linear com saturação

Tabela 2.2: Algoritmos de aprendizado.

Algoritmo	Descrição
trainlm	*Backpropagation* Levenberg-Marquardt
traingd	*Backpropagation* de gradiente decrescente
traingdm	*Backpropagation* de gradiente decrescente com momentum
traingda	*Backpropagation* de gradiente decrescente com taxa adaptativa
traingdx	*Backpropagation* de gradiente decrescente com momentum e taxa adaptativa

3. Definição dos parâmetros de treinamento

Tabela 2.3: Parâmetros de treinamento.

Parâmetro	Descrição
net.trainParam.epochs	Número máximo de épocas de treinamento
net.trainParam.goal	Erro desejado
net.trainParam.show	Número de iterações que aparece na tela
net.trainParam.lr	Taxa de aprendizado
net.trainParam.time	Tempo máximo (segundos) para o treinamento

4. Treinamento da rede

A função **train** é utilizada para treinar a rede de acordo com net.trainFcn e net.trainParam.

```
net = train(net, P, T)
```

5. Simulação da rede

   A função **sim** é utilizada para simular a rede.

```
Y = sim(net, P)
```

Exemplo

Dados os padrões de entrada e saída, P = [0 1 2 3 4 5 6 7 8 9 10] e T = [0 1 2 3 4 3 2 1 2 3 4], criar uma rede neural com o algoritmo de aprendizado *Backpropagation* sem realimentação.

A solução deste exemplo é descrito na Figura 2.5.

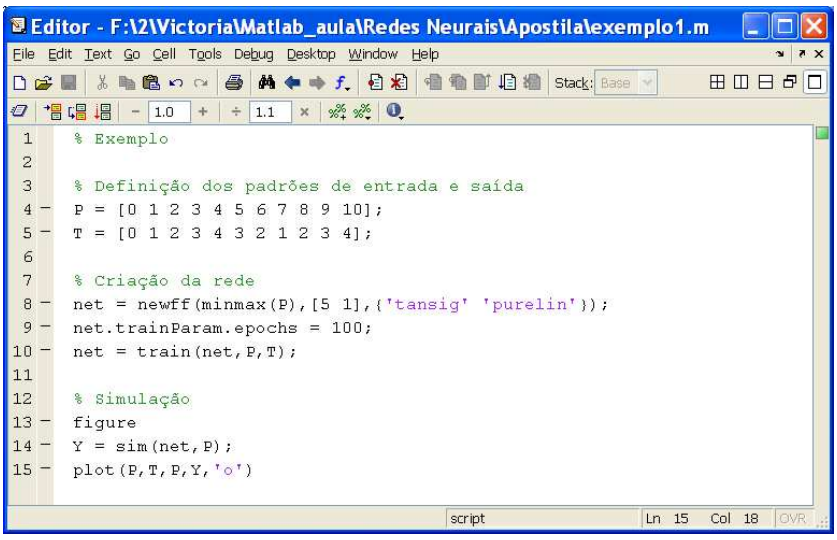

Figura 2.5: Exemplo de uma rede neural utilizando comandos num arquivo ".m".

A partir desses comandos, uma janela será gerada: *Neural Network Training (nntraintool)* (Figura 2.6), onde os parâmetros de treinamento da rede são mostrados, tais como números de camadas, algoritmo de treinamento, números de épocas, tempo de simulação e outros. Além disso, é possível plotar algumas características da rede ao longo de seu treinamento, como por exemplo, o desempenho da rede ilustrado na Figura 2.7.

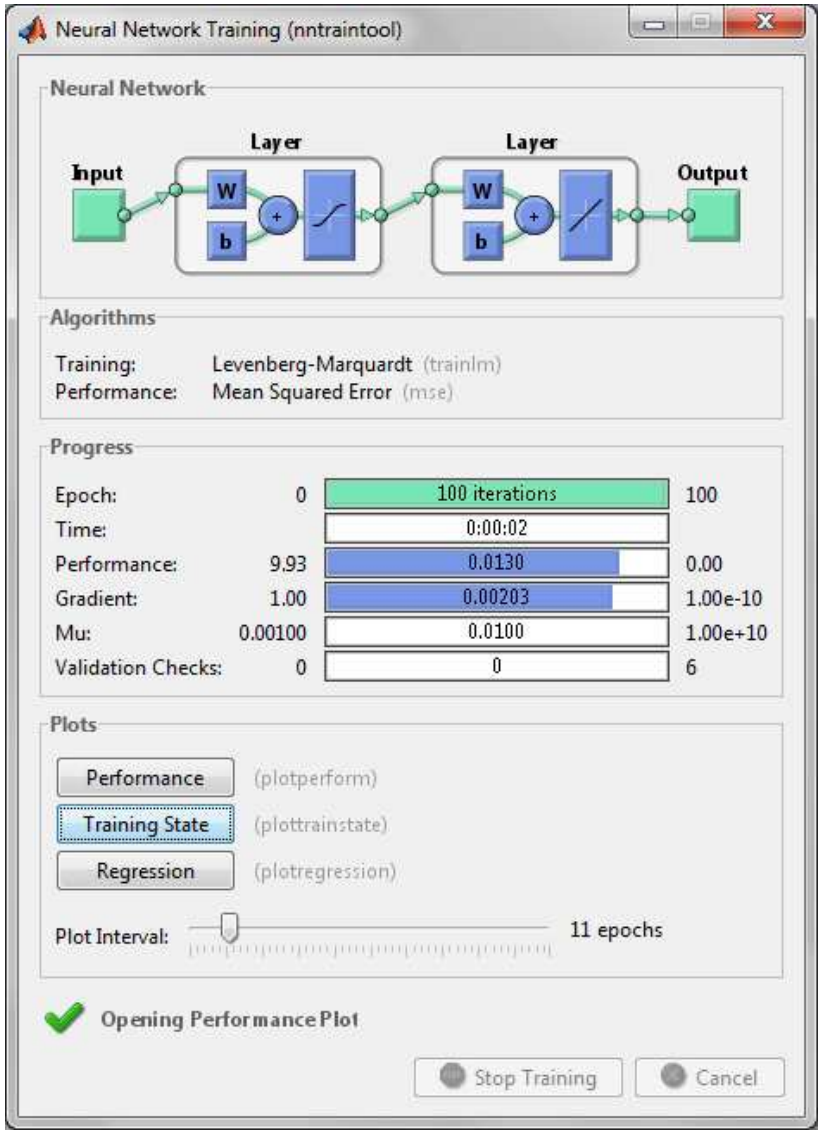

Figura 2.6: *Neural Network Training (nntraintool).*

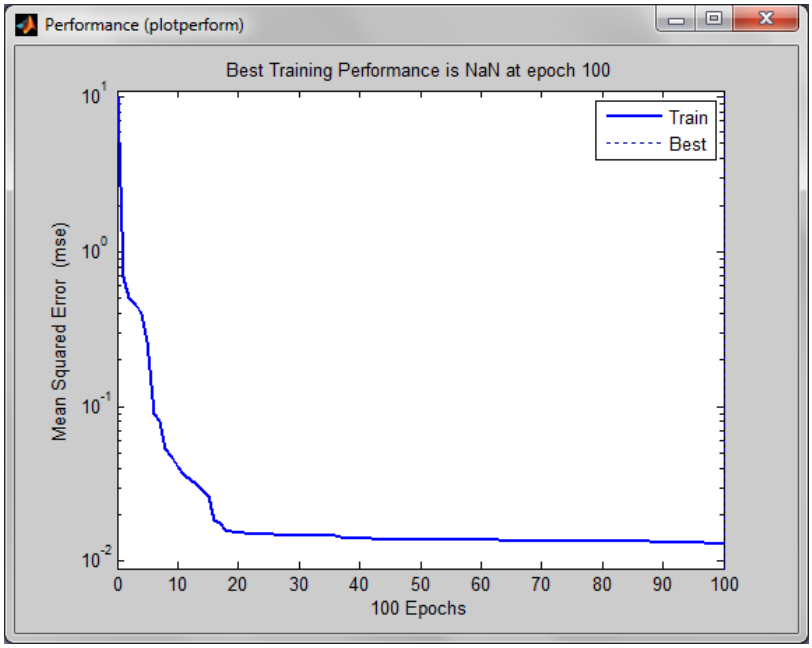

Figura 2.7: Treinamento da rede neural.

A comparação entre a saída da rede e a saída desejada é mostrada na Figura 2.8. Observa-se que a saída desejada (linha contínua) é muito próxima à saída da rede (círculos), indicando um bom desempenho da rede neural artificial.

Vale salientar que o desenvolvimento de uma RNA varia de acordo com o problema em questão e o objetivo de cada projeto. Se neste exemplo o objetivo fosse obter uma saída com um erro bem próximo de zero, alguns ajustes poderiam ser feitos para melhorar ainda mais esse desempenho.

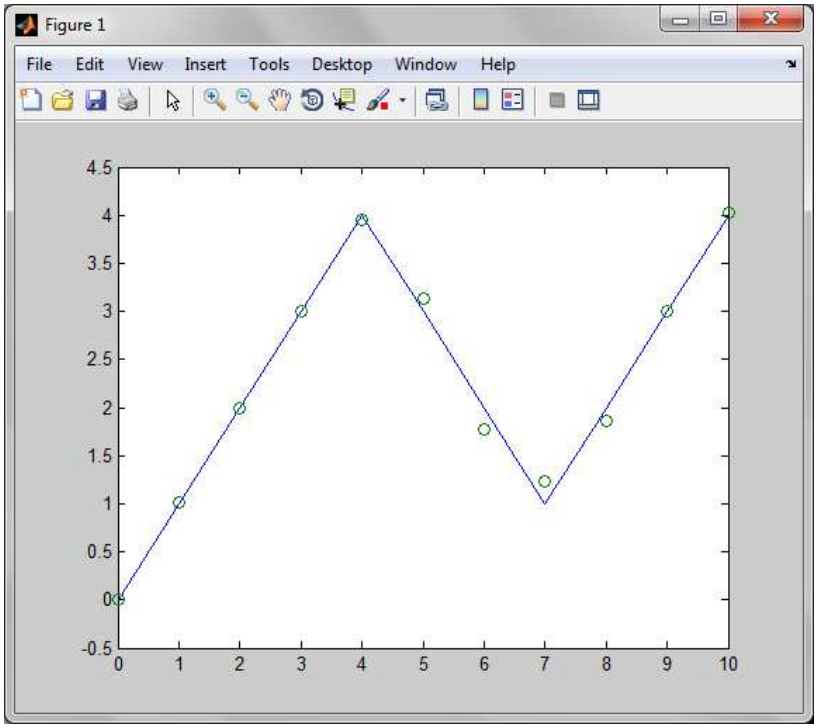

Figura 2.8: A comparação entre a saída da rede e a saída desejada.

Referências Bibliográficas

[1] S. Haykin, *Neural Networks: a comprehensive foundation.* New York: MacMillan College Publishing Co., 1999.

[2] G. Shepherd, *The synaptic organization of the brain.* New York: Oxford University Press, 2003.

[3] M. Roberts, "Schizophrenia daily news blog: New neurobiology knowledge may lead to schizophrenia research and treatments," *Disponível em: http://www.schizophrenia.com/sznews/archives/005490.html*, 2007.

[4] W. McCulloch and W. Pitts, "A logical calculus of ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.

[5] D. E. Rumerlhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors, 1986," *Nature*, vol. 323, pp. 533–536, 1986.